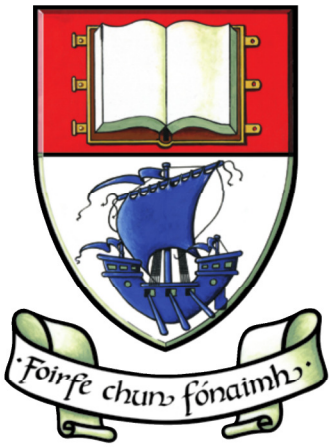


Waterford Institute *of* Technology

INSTITIÚID TEICNEOLAÍOCHTA PHORT LÁIRGE

PROJECT
HIGHER DIPLOMA IN SCIENCE
IN COMPUTER SCIENCE (NFQ – LEVEL 8)

2013-2014

ANAEROBIC DIGESTION CALCULATOR

FINAL REPORT

by

Patrick Murphy

Supervisor: Eamonn de Leastar

School of Lifelong Learning and Education

Declaration of Authenticity

I declare that the work which follows is my own, and that any quotations from any sources (e.g. books, journals, the internet) are clearly identified as such by the use of ‘single quotation marks’, for shorter excerpt and identified italics for longer quotations. All quotations and paraphrases are accompanied by (date, author) in the text and a fuller citation is the bibliography. I have not submitted the work represented in this report in any other course of study leading to an academic award.

Student..... Date .....

Word Count

This project contains 6179 words.

Statement of Copyright

Abstract

The problem the application is attempting to solve is to develop an app that calculates the energy output from biogas generated through Anaerobic Digestion (AD). AD is a process where bacteria break down organic materials in the absence of oxygen into biogas and digestate. The biogas can be used to produce electricity and heat in a Combined Heat and Power Plant, or purified and used as a vehicle fuel or in the gas grid. AD is a carbon neutral process. Calculations include biogas output, methane output, heat output, electricity output, purified biogas volume, number of vehicles that could be fuelled, potential income and carbon dioxide savings from the various biogas end uses. The algorithms to calculate energy output form AD are not easily accessible to the general public.

The problem was broken down into user stories. Each user story was assigned to an iteration; the project was divided into four iterations. Criteria were formulated to confirm that each user story had been completely successfully. The software was designed as a web app so it would be available on the maximum number of devices. The app has two types of user a normal user and an admin user. The normal user can sign up to the app, create a project with multiple calculations, and view project calculation results. An admin user can add organic materials (feedstock's), grant admin access, and view user calculations. The app has four models feedstock, calculation, project and user. A user has many projects, a project has many calculations, and a feedstock has many calculations. When a project is created the data is saved to the relevant database tables. This data is retrieved from the database, run through the AD algorithms in the calculation and project models; the data from the algorithms is displayed in chart format on the various calculation pages. Behaviour driven and manual testing was used to test the app.

The aim of the project is to turn an Excel mathematical model developed by the author that calculates energy output from AD into a user friendly web app. The app was built with the Ruby on Rails framework. The RSpec testing tool was used to test the app; test data was set up with Factory Girl. Bootstrap 3 was used for to style the app. The app is hosted on the on the Heroku cloud service platform.

Keywords

Ruby on Rails; Web App; Dynamic; Anaerobic Digestion Calculator;

Acknowledgements

I would like thank my supervisor Eamonn de Leastar for his guidance and advice.

I would also like to thank my Work Place Mentor John Stacey and rest of the Betapond staff that helped my with my first Ruby on Rails project.

Table of Contents

Declaration of Authenticity ii

Word Count ii

Statement of Copyright ii

Abstract iii

Keywords iii

Acknowledgements iv

Table of Contents iv

List of Figures vi

Chapter 1 Introduction 1

1.1 Goals 1

1.2 Method 1

1.3 Report Overview 2

Chapter 2 Background and Problem Statement 3

2.1 Introduction 3

2.2 Review 3

2.3 Problem Statement 4

Chapter 3 Project Structure 5

3.1 User Stories 5

3.2 User Case Diagrams 8

3.3 Project Management 8

Chapter 4 User Manual 10

4.1 Overview 10

4.2 System 10

4.2.1 Login and Signup 10

4.2.2 Create a Project and View Calculation Pages 13

4.2.3 Admin User 18

Chapter 5 Software Design 22

5.1 Introduction 22

5.2 High Level Design 23
5.3 Detailed Design 24
5.3.1 Design Features 25
5.4 Data Storage 28
5.5 Verification 28
5.6 Validation 30
Chapter 6 Discussion and Conclusion 31
6.1 Project Review 31
6.2 Key Skills 31
6.3 Future Work 31
6.4 Conclusion 32
Bibliography 33
Appendices 35
Appendix A 35
A1 Static Website 35
Appendix B 36
B1 Java Models and JUnit Tests 36

List of Figures

Figure 1 User Case Diagram 8
Figure 2 Admin User Case Diagram 8
Figure 3 Project Plan Gantt Chart 9
Figure 4 Home Page 10
Figure 5 Signup Page 11
Figure 6 Login Page 11
Figure 7 Project List 12
Figure 8 Edit User Details Page 12
Figure 9 New project page 13
Figure 10 Dynamic form 13
Figure 11 Dropdown list 14
Figure 12 Iteration 4 Chart display 14
Figure 13 Chart Information. 15
Figure 14 Popup Information 15
Figure 15 Calculation Links 16
Figure 16 More Info Button 16
Figure 17 Project Report Page 17
Figure 18 Print Report 17
Figure 19 Admin Navigation Tab 18
Figure 20 Admin Dashboard Page 18
Figure 21 Feedstock List 18
Figure 22 Edit a Feedstock Page 19
Figure 23 User List Page 19
Figure 24 Add a new Feedstock Page 19
Figure 25 Edit User Page 20
Figure 26 User Project Page 20
Figure 27 User Project Calculations Page 20
Figure 28 User Project Individual Calculation Page 21
Figure 29 Model, Controller, and Views 23
Figure 30 Models and Controllers 24
Figure 31 Upgrading is feasible Message 27
Figure 32 Data Storage 28
Figure 33 Model tests 29

Chapter 1 Introduction

This chapter introduces the project. The following areas are covered: listing of project goals, description of why the problem needs to be solved, brief description of how the project was completed, a summary of algorithms used, and a description of the report structure.

A repository of the project is available at: https://github.com/PatMurp/adapp.git.
The project URL is: http://adcalapp.herokuapp.com/

1.1 Goals

The following are the project goals:

1. Convert an Excel mathematical model designed by the author to calculate the energy output from organic materials using Anaerobic Digestion (AD) into user friendly web app.
2. Allow a user with limited knowledge of AD to be able to use the app; the user should be able to select from a dropdown list of organic materials and press a calculate button.
3. Use charts to display calculation output and allow a user to print a calculation report.
4. Use the web app to educate the user to the economic and environmental benefits of using AD for energy production.

The motivation behind the project is that the algorithms used to calculate the energy output from (biogas) AD are only available in various academic journals and are not easily accessible by the general public. The app allows a user to use these algorithms to complete various calculations.

1.2 Method

A static website was created to help the author design the app layout (Appendix A1). It includes a home page and various calculation result pages.

A Java project was created, the author is familiar with the Java language, to turn the Excel calculations into a series of Java models (Appendix A2). The models simulated the main processes that were planned for the app. JUnit tests were written to test the various models.

It was decided to build the app in Ruby on Rails as it is the framework used by Betapond the company where the author was on work placement. The project was broken into four iterations.

1. Calculation with hardcoded values.
2. Calculation with dynamic values.
3. Calculation with multiple feedstock's'.
4. Improve user interface and add an admin user.

The algorithms in the app calculate: the amount of energy that can be generated form the selected organic materials, the volume of material remaining after AD and the volume of nutrients in this material, the amount of heat and electricity can be generated from the biogas and its potential value, the volume of biogas available after purifying and its potential value, the number of vehicles and households that the purified biogas could fuel, and the carbon dioxide (CO_2) savings from the various biogas end-uses.

1.3 Report Overview

Chapter 1 introduces the project. Chapter 2 provides an explanation to the problem the app is solving. Chapter 3 outlines the project management and problem analysis. Chapter 4 provides instructions on how to use the software. Chapter 5 describes the software design and implementation. Chapter 6 evaluates the project and present the conclusions drawn from the project.

Chapter 2 Background and Problem Statement

This chapter describes the problem outlined in the introduction. It provides a brief description of the AD process, possible uses for biogas, reviews similar web apps in this area, reviews technologies considered for the app, and states the problem the app is attempting to resolve.

2.1 Introduction

Anaerobic Digestion is a process where biogas is produced from the breakdown of organic materials by bacteria in the absence of oxygen; it occurs naturally in cow stomachs, peat bogs, and landfill sites. This process is carried out is a digester; these can range in size from a household digester, typically found in Asia, to an industrial scale plant.

Organic materials that can produce biogas include: animal slurries, industrial bio-wastes, organic waste from food industries, sewage sludge, food waste, and energy crops (e.g. grass silage and maize). When the organic materials leave the digester they retain any nutrients that were available when they entered the digester; this material is called digestate and can be used as a replacement for artificial fertilizers.

When the biogas is produced it can be used in a Combined Heat and Power Plant (CHP) to generate electricity and heat. In the Republic of Ireland tariffs are available for electricity produced from CHP. If an outlet can be found for the heat it increases the profitability of the plant.

The biogas can be purified (upgraded) and used as a vehicle fuel or injected into the gas grid. Biogas is carbon dioxide (CO_2) neutral; the CO_2 omitted when the biogas is burned was captured when the organic material was growing. Carbon credits can be generated with AD.

2.2 Review

At the time of writing (April 2014) there is only one free to use AD calculator available online [1]. It is in beta phase at the moment. It allows a user to select tariffs from several countries to calculate potential income; the Republic of Ireland is not included.

Ruby on Rails

Ruby on Rails (Rails) is an open source web application framework written in the Ruby language. It uses the Model Controller View (MVC) pattern. A model in a Rails project maps to a database table by default. A Controller responds to requests from the web server; if required save or fetch data from a model, and uses a view to create a HTML page [2]. RESTful routes are used for external web requests; the default route actions are create, new, edit, update, destroy, show, and index. A view uses Embedded Ruby (EMB) to dynamically generate webpages. To use third-party libraries in Rails a gem is used; gems are added to a project by adding the gem name to a file called gemfile [2].

Devise Gem

Devise is an authentication gem for Rails projects. It encrypts and stores a user password in a database which is used to authenticate a user when signing in, it validates emails and user passwords, expires sessions that have no activity for a set time, tracks number of sign ins, tracks timestamps, tracks IP address, allow a user to edit their email and password, and destroy a user account [3].

Active Admin Gem
The Active Admin gem is used to create administration interfaces. It allows interfaces to be created that enable an admin user to manage data. The user interface can be easily customised. When an admin user logs in an admin dashboard displays a navigation link for any models registered with active admin. Model relationships are detected. Active Admin allows an admin user to download data in CSV, XML, or JSON formats [4].

ChartKick Gem
The ChartKick gem allows a user to generate dynamic charts using Google Charts API or HighCharts. Chart types that can be created include: line charts, pie charts, column charts, bar charts, area charts, and geo charts. Data can be passed into the chart as a hash or an array [5].

jQuery
jQuery is an open source JavaScript library that enables HTML document navigation, document manipulation, event handling, animation, and Ajax in multiple web browsers. jQuery plugins can be used to increase its functionality. jQuery can be used in a web page by linking to a local copy of the library or by a linking to a copy on a public server [6].

RSpec & Factory Girl
RSpec is a behaviour driven development tool for the Ruby language. An RSpec test is enclosed in a describe block. A context method allows the test to be broken into human readable logical steps. An RSpec test contains an executable example of the expected behaviour [7]. Factory Girl is a library for setting up test data. The advantage of Factory Girl is that it is not fixed to a specific phase of development; if new attributes are added Factory Girl will see them [7].

Heroku
Heroku is a Platform as a Service (PaaS); it allows a developer to push code and basic configuration to get an application running. Languages supported include Ruby, Java, Python, Node.js, Scala, and PHP. An application must be committed to a Git repository before it can be deployed to Heroku.

Git
Git is an open source version control system used to keep track of different versions of files in a software project. The init command is used to create a new Git repository. The clone command creates a copy on an existing copy. The add command moves changes from a working directory to a staging area. The commit command takes the changes in the staging area and saves them in the project history.

2.3 Problem Statement

Develop a user friendly web app to calculate the energy available from various organic waste materials through AD. Compare the different end uses of biogas using potential income and CO_2 savings.

Chapter 3 Project Structure

This chapter describes the project structure. User stories and User Case Diagrams were used to define the tasks that the user needs to preform to solve the problem. It outlines how the project was planned and why this method was chosen.

3.1 User Stories

User stories were used to define the features that are required from a user perspective. Two types of user can use the app a normal user and an admin user. As project progressed user stories that were not completed in a single iteration were broken into smaller stories. The user stories were assigned to an iteration and acceptance criteria were formulated which needed to be confirmed to finish the story.

User stories

I want to enter a volume for a hardcoded organic material and press calculate a button
so that I can view the biogas output from that material

Priority iteration 1
Acceptance Criteria
- Generate a biogas value
- Confirm that volume value is correct
- Confirm that biogas value is correct
- Pass calculation method test

I want to select from a list of pre-populated organic material data
so that I can easily create a calculation

Priority iteration 2
Acceptance Criteria
- item can be selected from drop down menu
- Confirm that selected item is displayed
- Confirm that calculation values are correct
- Pass calculation methods tests

I want to sign up and login into application
so that I can save and access my calculations

Priority iteration 3
Acceptance Criteria
- User can signup or login
- User has to login or signup to use app
- Confirm user can only view their calculations
- Confirm that saved calculations are available

I want to have the ability to edit calculations
so that I can modify incorrect data and add or delete feedstock's

Priority iteration 2
Acceptance Criteria Edit page is available
Selected calculation is editable
Confirm calculation result changes if values are edited

I want to have the ability to add multiple feedstock's to a calculation
so that I can calculate energy output from a group of feedstock's

Priority iteration 3
Acceptance Criteria Confirm that additional fields can be added
Confirm that multiple fields calculations result are displayed
Confirm that values and totals are correct

I want to display calculation results in graphs
so that I can get a visual representation of the data

Priority iteration 4
Acceptance Criteria Graph is displayed
Confirm that values in graphs are correct

I want to view data in table format if required
so that I can view detailed calculation data

Priority iteration 4
Acceptance Criteria Confirm that table not displayed by default
Confirm that table is displayed when show button is selected
Confirm that table is hidden when hide button is selected

I want to display a project report
so that I can view all information about a calculation

Priority iteration 4
Acceptance Criteria Confirm that project report is displayed
Confirm report values are correct for selected project

I want to print the report
so that I can have a hard copy of project on file

Priority iteration 4
Acceptance Criteria Confirm that correct values are displayed
Confirm that calculation results are on separate pages

Admin User Stories

I want to have the ability to add and edit feedstock data
So that I can add and update data as it becomes available

Priority iteration 4
Acceptance Criteria Confirm that user can view all feedstock's
Confirm that the add feedstock page is displayed
Confirm that a new feedstock can be added
Confirm that a user can edit feedstock values

I want to have the ability to grant admin access to a user
So that I can have multiple admin users

Priority iteration 4
Acceptance Criteria Confirm that a user does not have admin access
Give this user admin access
Confirm that the user now has admin access

I want to have the ability to view user projects
so that I can monitor user activity

Priority iteration 4
Acceptance Criteria Confirm that a user can view a user's details
Confirm that a user can view a user's projects
Confirm that a user can view project calculations
Confirm that a user can view calculation values

3.2 User Case Diagrams

A user can sign up or login to the app. A user can cancel their account. When logged in a user can create a new project or view existing projects. When creating a new project a dynamic form is used; additional fields can be added to the form. When viewing a project the user can view the various calculation results on separate pages, view additional information with popup boxes, and print a report of all project calculations. A user can edit or delete a project.

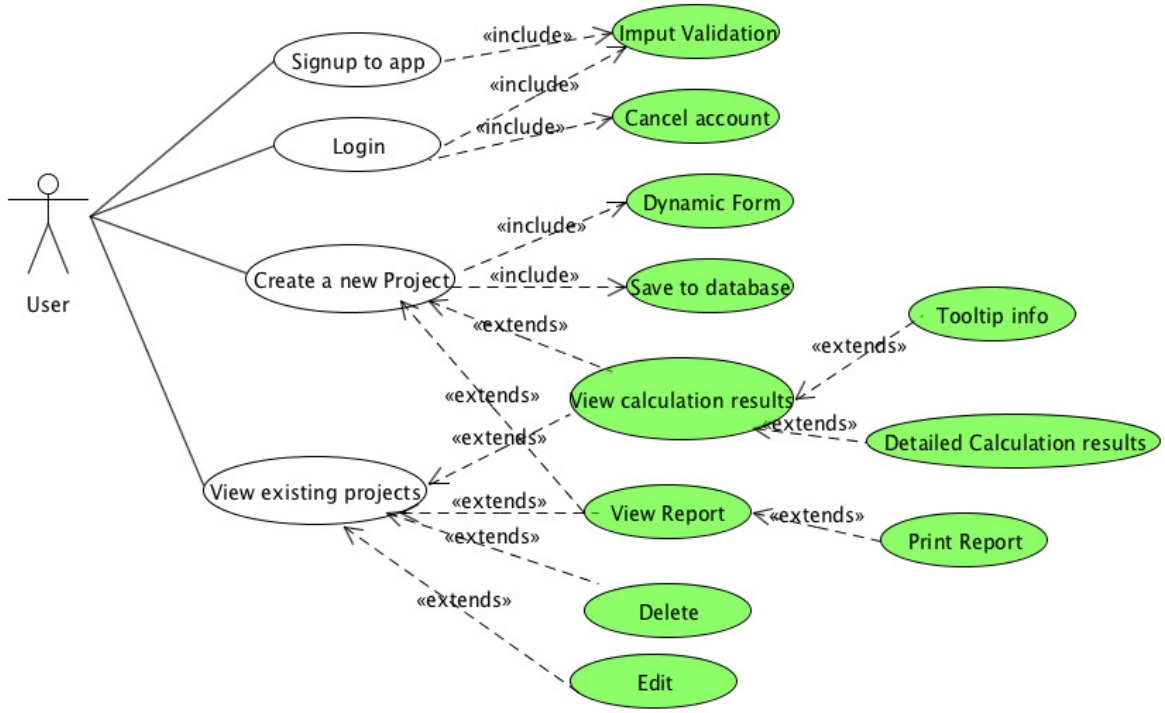


Figure 1 User Case Diagram

An admin user can grant admin access to an existing user. They can also add, edit, or delete a user. An admin user can add a new feedstock or edit existing values. An admin user can view a user's calculation data.

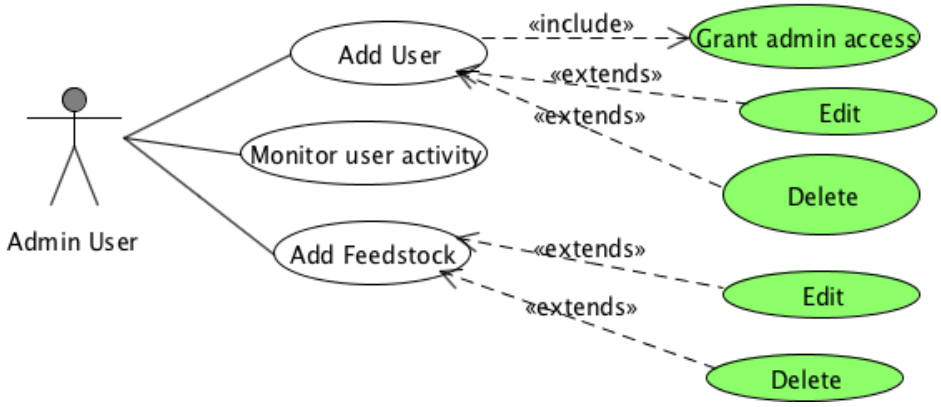


Figure 2 Admin User Case Diagram

3.3 Project Management

The project was divided into four iterations. This approach was taken because it enabled the problem to be broken into smaller portions that were easier to solve; it enabled improvements to be made as features were added.

In the initial project plan there were four iterations each lasting 21 days and an 8 day period at the end to test the UI and complete the project report. Initially there was a steep learning curve with Ruby on Rails which delayed completion of Iteration 1. Iteration 2 went according to plan and the days lost in iteration 1 were made up. Iteration 3 and 4 went better than expected and were completed in 4 weeks rather than the allocated 6 weeks.

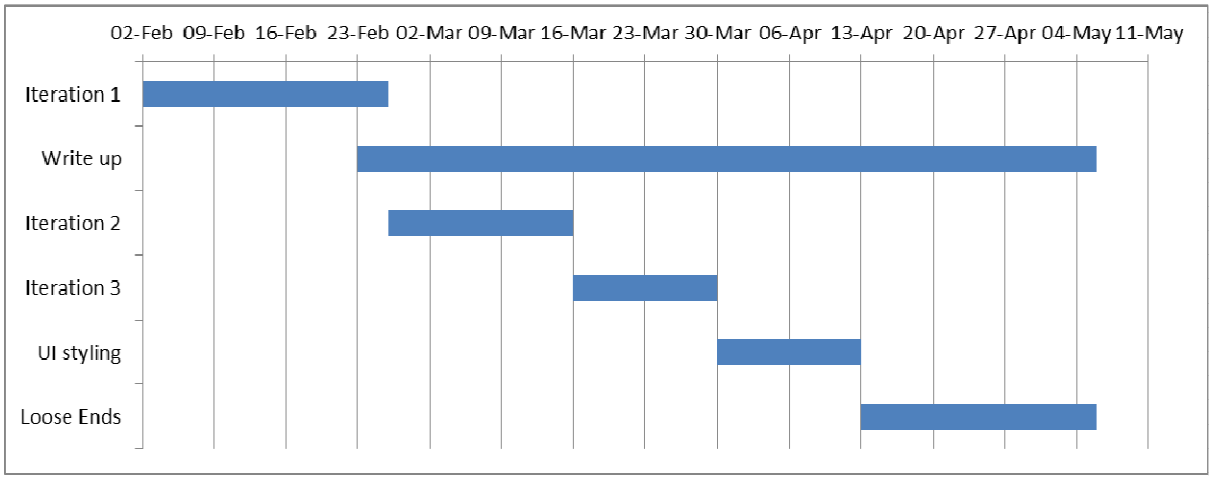


Figure 3 Project Plan Gantt Chart

Chapter 4 User Manual

4.1 Overview

This application is a web app that calculates the potential biogas and methane from various organic materials. A user can create a project containing multiple waste materials. The application saves this data to a database. This data is then retrieved from the database and the application preforms various calculations to display different calculation result pages. The app was designed for PC's and Tablets, CSS styling was added to display on mobile devices.

4.2 System

There are three main activities that can be performed; a user can sign up to the app, a logged in user can create a new project and view calculation pages or view calculation pages for an existing project, an admin user can create new feedstock's, grant admin access and monitor user activity.

4.2.1 Login and Signup

The URL for the app is http://adcalapp.herokuapp.com/. As the app is hosted on Heroku free apps there can be up to a 20 second delay in displaying the home page.

The home page has options to login or signup via the navigation bar or via signup and login buttons.

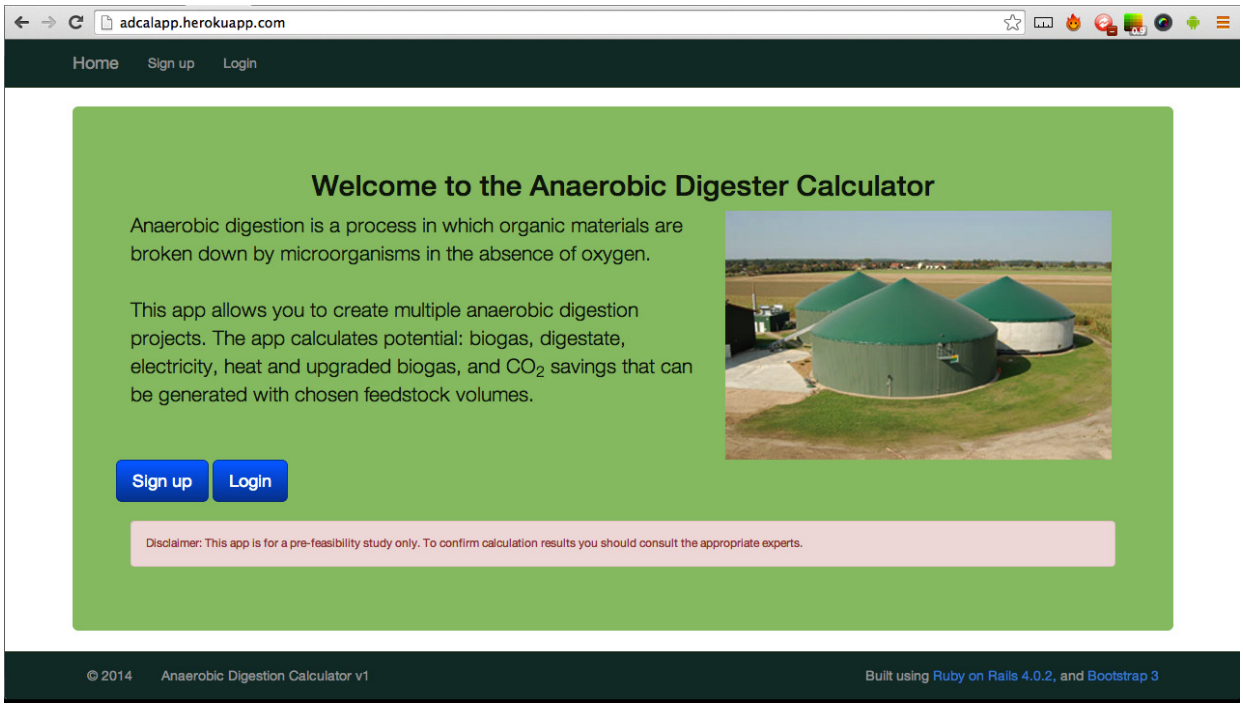


Figure 4 Home Page

The signup page requires the email, password, and password confirmation fields to be filled in. Ensure that a valid email is entered and that the password has at least eight characters. Press the signup button to signup; if already signed up press the login button.

Figure 5 Signup Page

The login page requires the email and password fields to be filled in. Only use the login page if you have already signed up otherwise an invalid error message will appear at the top of the page. Press the login button to login.

Figure 6 Login Page

The View Projects page is displayed after login. Additional tabs are added to the navigation bar after a successful login. The Create New Project link will display the new project page. Edit user profile displays the user edit page, and the logout tab logs out of the application.

For each project listed there are four buttons that can be pressed; view biogas output displays the biogas page, project report displays the project report, edit displays the edit page, and the destroy button deletes a project. A popup confirmation message is displayed when destroy is selected, press ok if you want to delete a project.

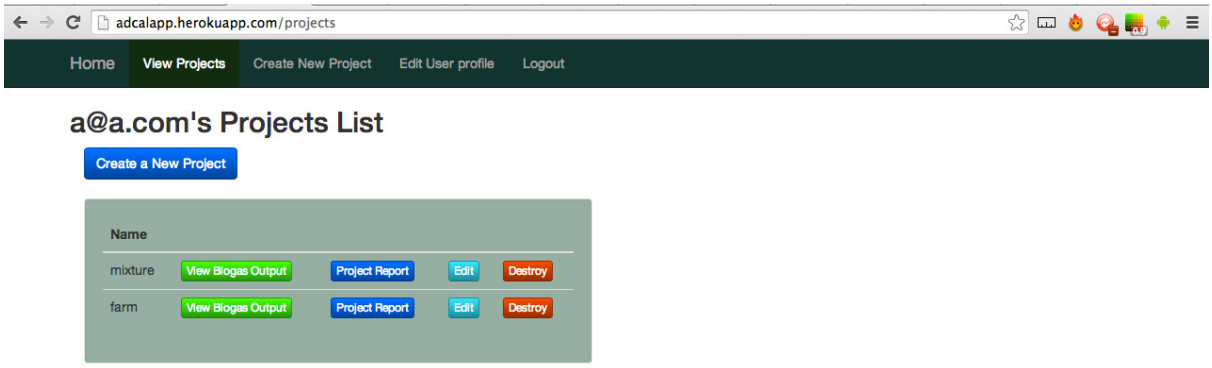


Figure 7 Project List

The edit user page allows a user to change their email and or password. The current password field has to be filled in to make any changes. A cancel my account button deletes the user from the database.

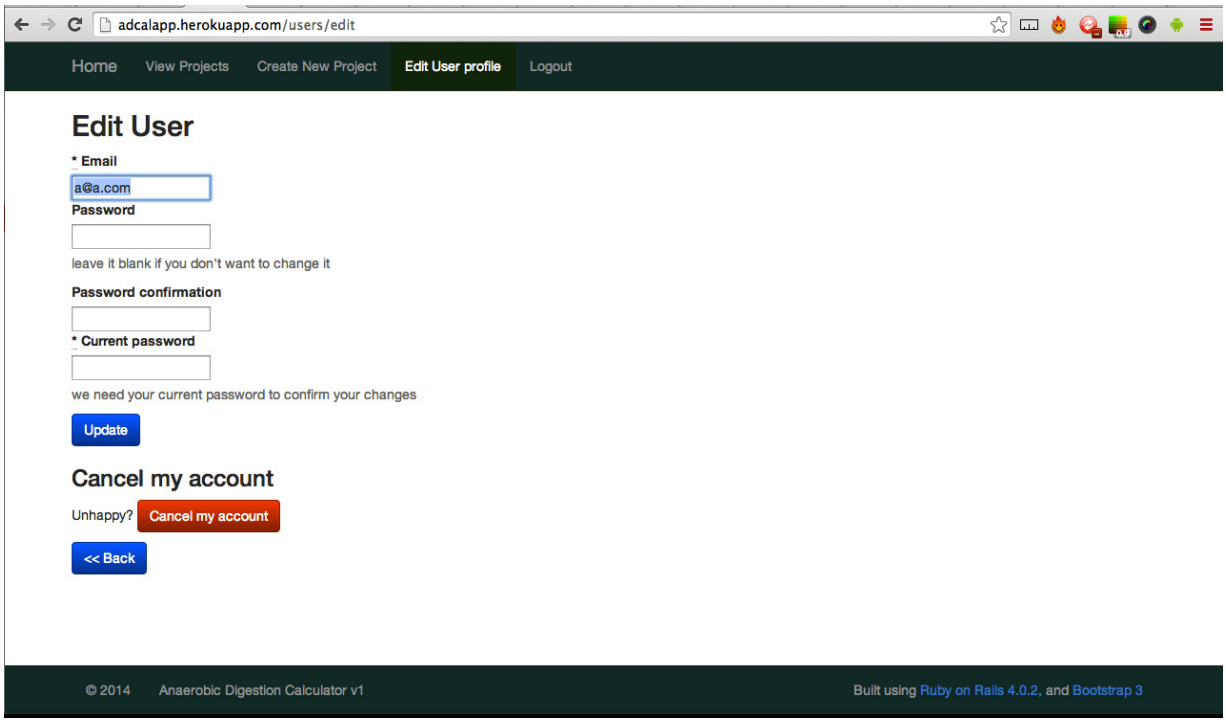


Figure 8 Edit User Details Page

4.2.2 Create a Project and View Calculation Pages

When the Create a New Project button or link is selected the new project page is displayed. Enter a project name, enter a positive number for tonnes, and select an organic material from the drop down list. HTML 5 validations are used to ensure that when a user is creating a new project that all fields are filled and that an item is selected from dropdown list. In the tonnes field negative numbers are not allowed.

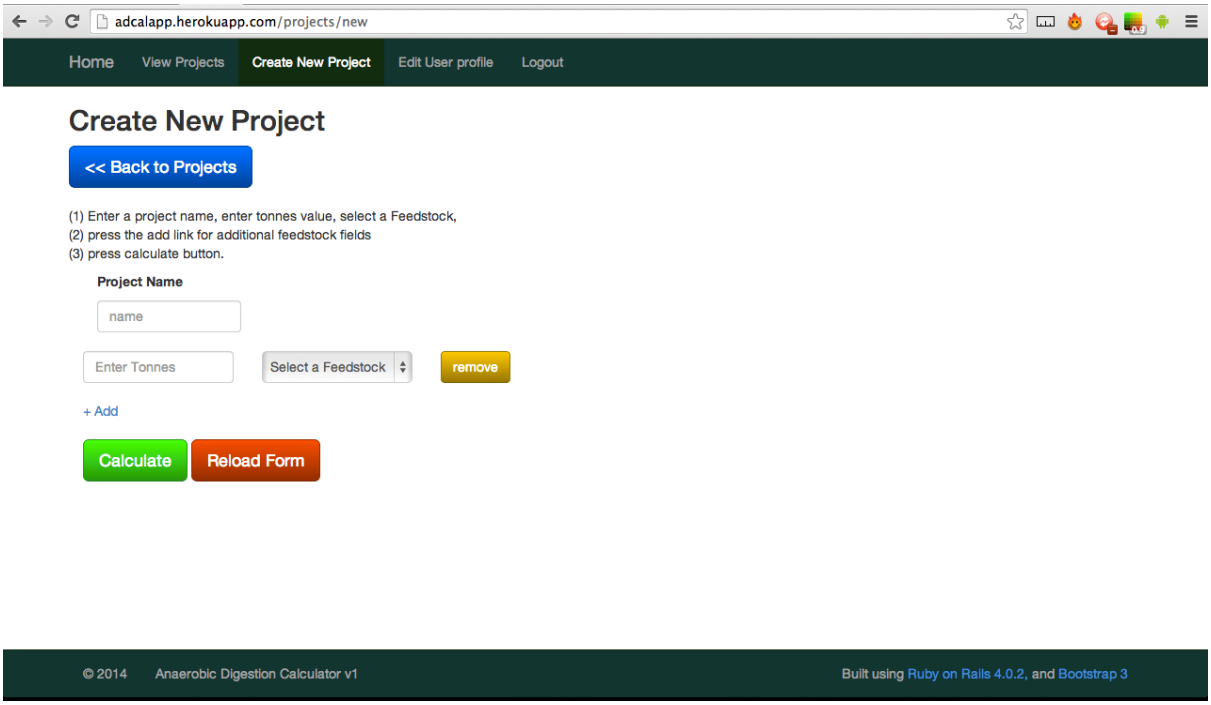


Figure 9 New project page

Press the add link to add an additional fields to the new project page. Press the Reload Form button to reload the default form with a single calculation field. Press the calculate button to save the values to a database and view the biogas output from the selected feedstock's.

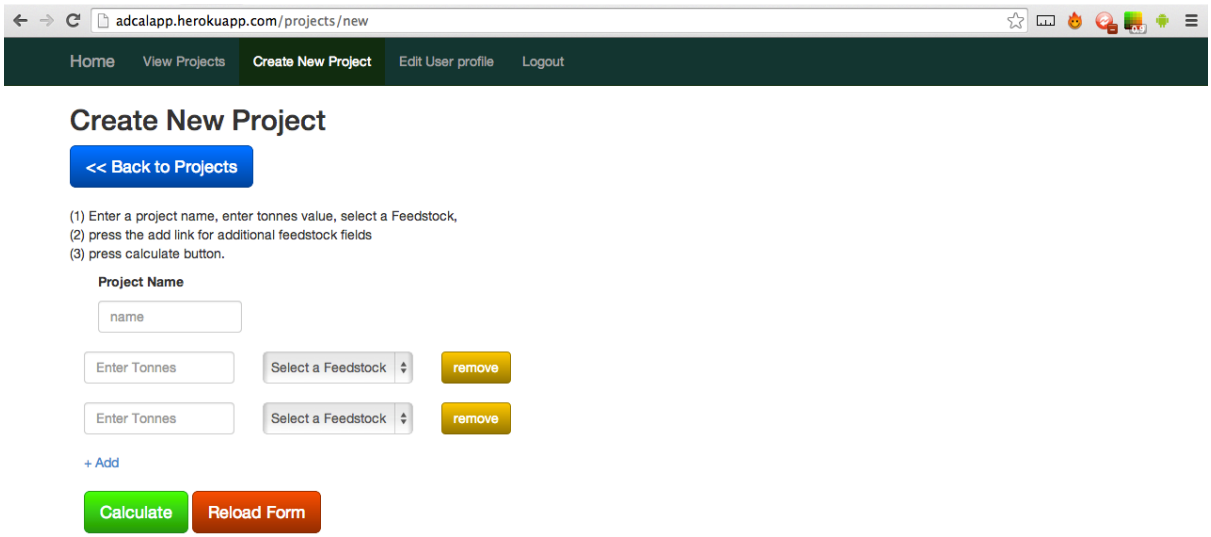


Figure 10 Dynamic form

The drop down list contains a pre-populated list of available feedstock's.

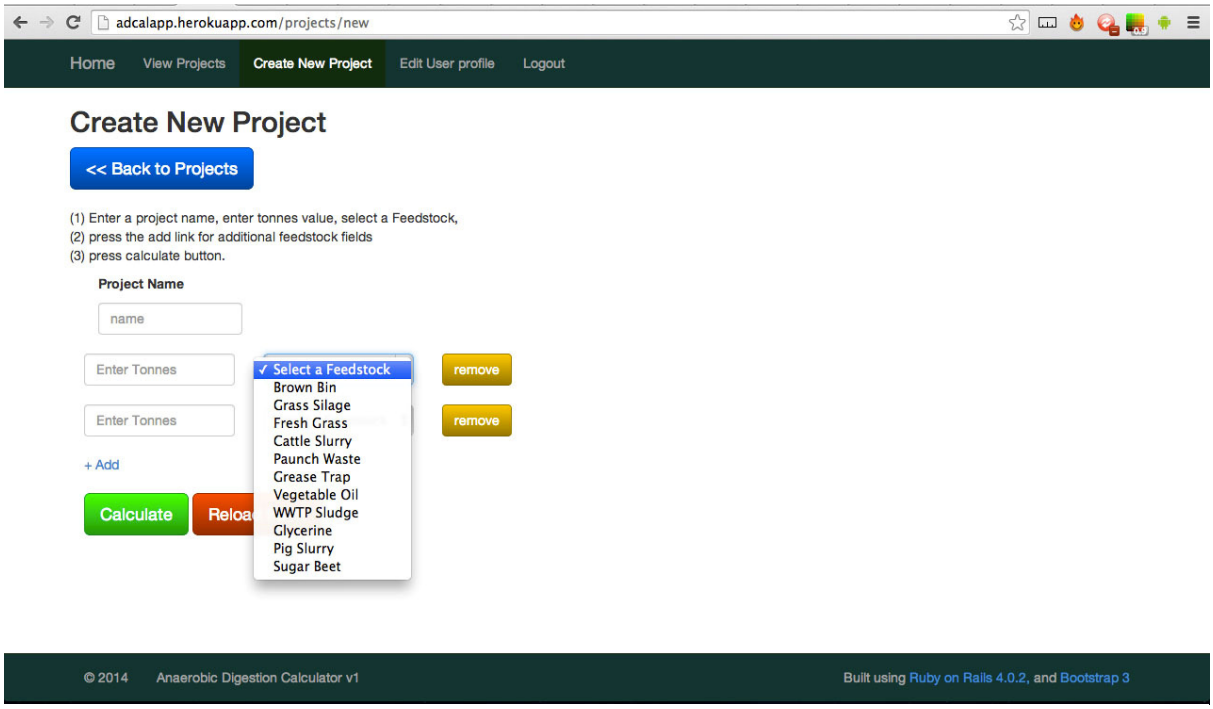


Figure 11 Dropdown list

The biogas page displays the potential biogas and methane output. Pie charts display the percentage contribution each feedstock makes to tonnes, biogas output, and methane output.

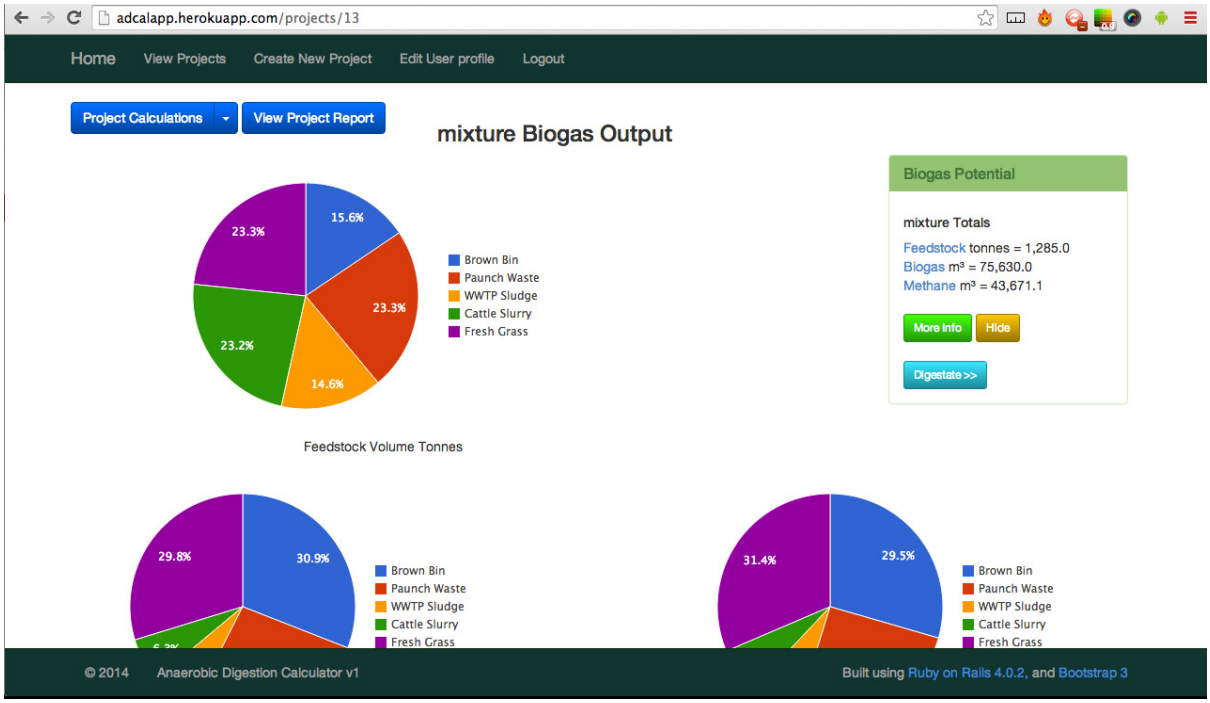


Figure 12 Iteration 4 Chart display

Hovering over a pie chart displays the feedstock name, volume, and percentage.

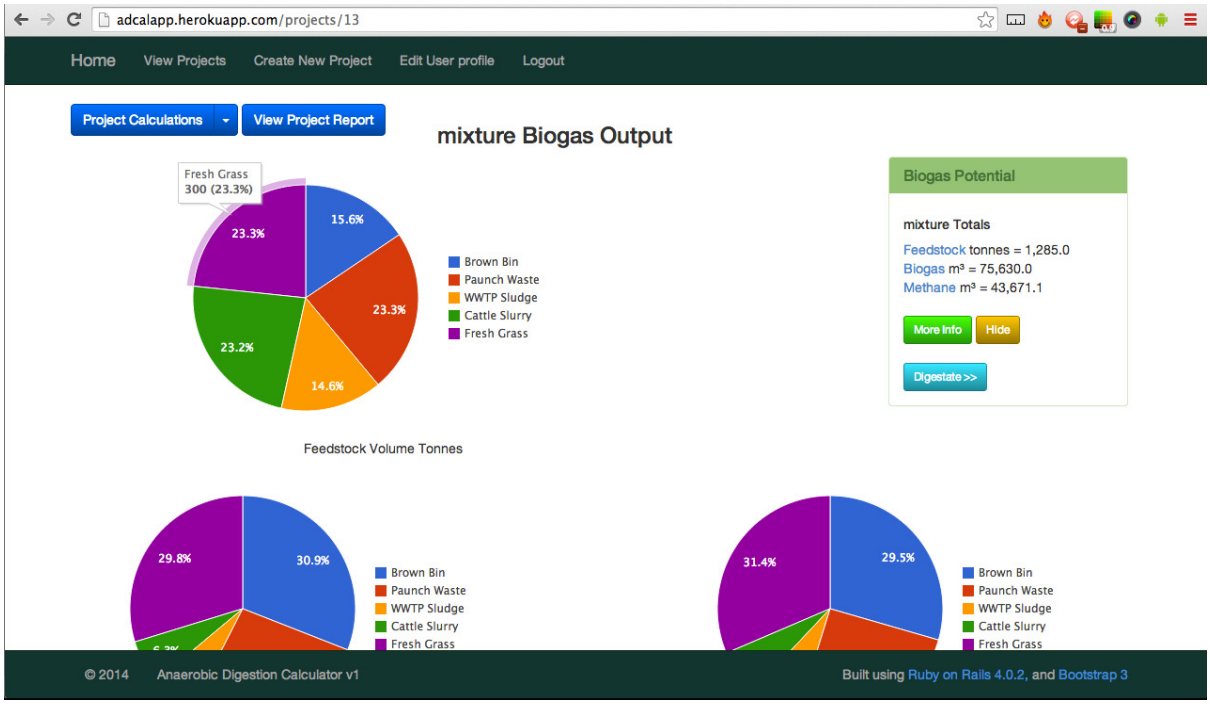


Figure 13 Chart Information.

When a link in the right hand side panel is selected additional information is displayed.

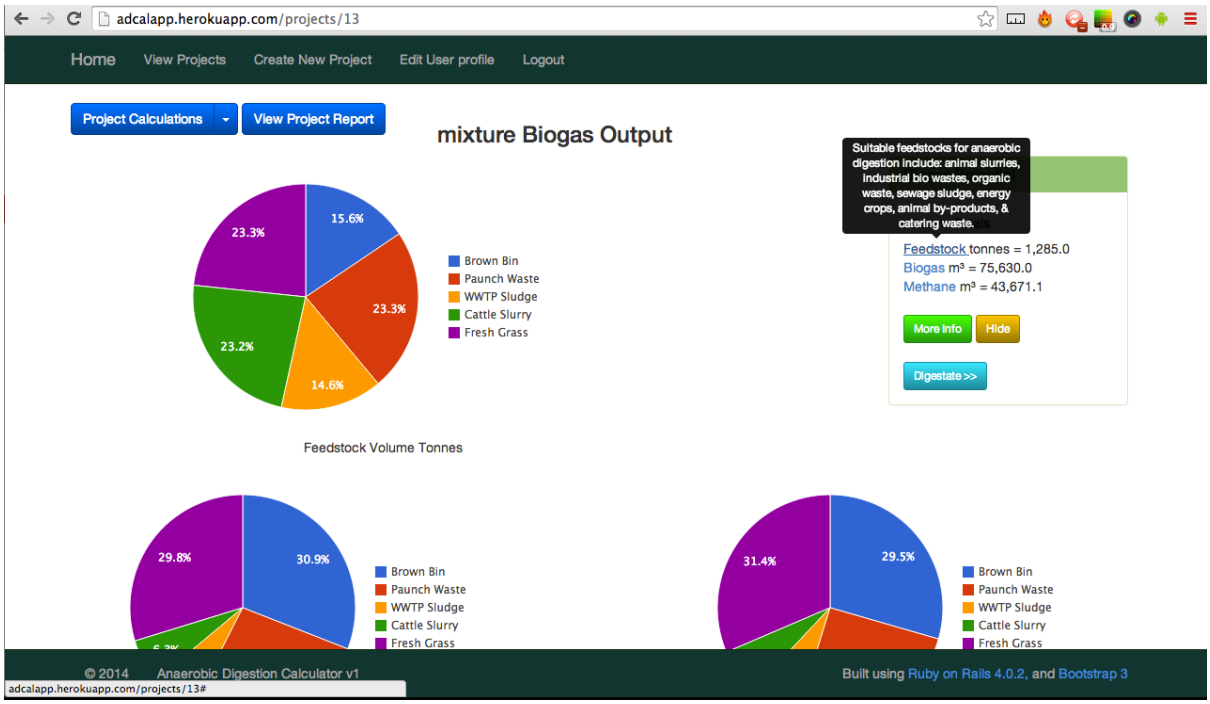


Figure 14 Popup Information

Links to other calculation pages are available from a the Project Calculations drop down buttton and via buttons in the right hand panel.

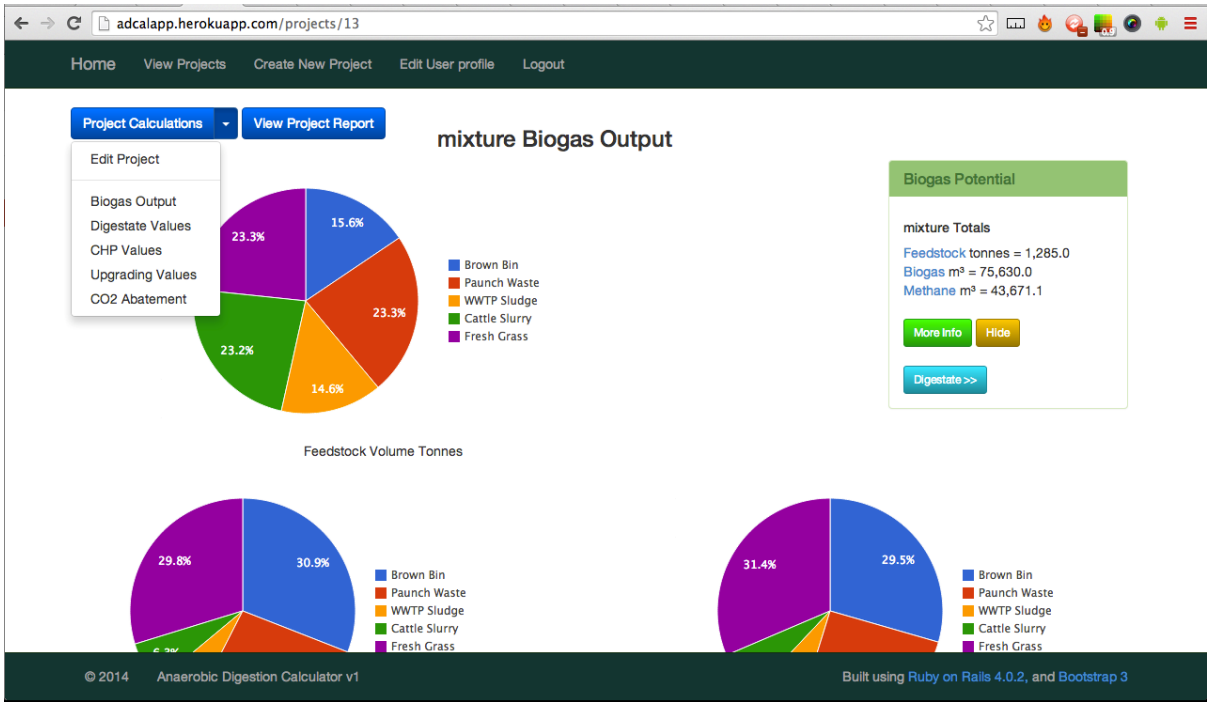
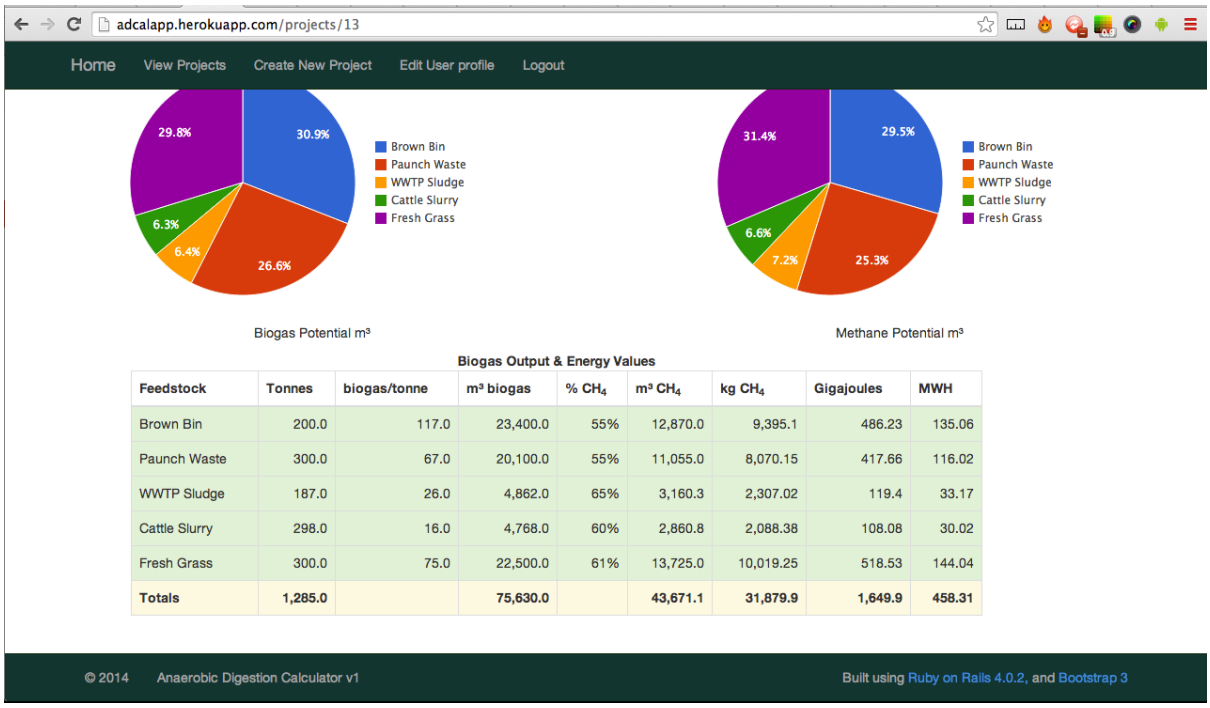


Figure 15 Calculation Links

The more info button displays the calculation data in table format. The hide button hides the table.



Biogas Output & Energy Values

Feedstock	Tonnes	biogas/tonne	m³ biogas	% CH_4	m³ CH_4	kg CH_4	Gigajoules	MWH
Brown Bin	200.0	117.0	23,400.0	55%	12,870.0	9,395.1	486.23	135.06
Paunch Waste	300.0	67.0	20,100.0	55%	11,055.0	8,070.15	417.66	116.02
WWTP Sludge	187.0	26.0	4,862.0	65%	3,160.3	2,307.02	119.4	33.17
Cattle Slurry	298.0	16.0	4,768.0	60%	2,860.8	2,088.38	108.08	30.02
Fresh Grass	300.0	75.0	22,500.0	61%	13,725.0	10,019.25	518.53	144.04
Totals	**1,285.0**		**75,630.0**		**43,671.1**	**31,879.9**	**1,649.9**	**458.31**

Figure 16 More Info Button

When the View Project Report button is pressed the report page is displayed. It displays all calculation data for a project.

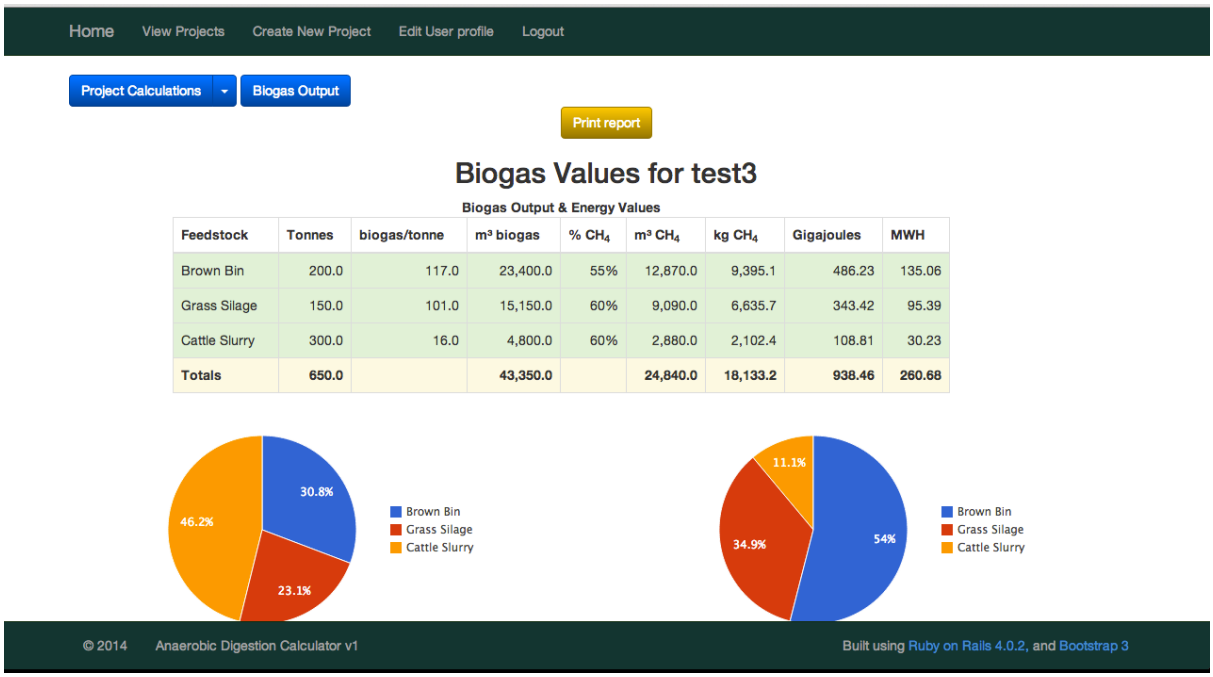


Figure 17 Project Report Page

When the Print Report button is pressed the calculation data is formatted to display on separate pages.

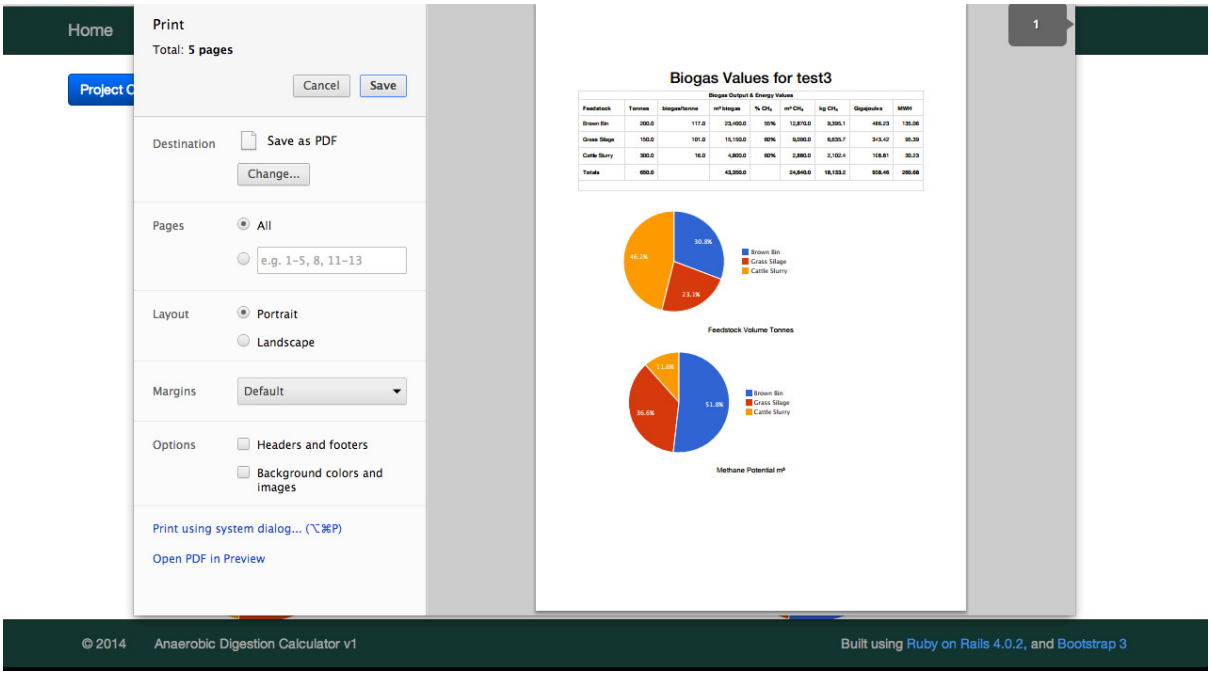


Figure 18 Print Report

4.2.3 Admin User

If a user has admin access an additional tab appears in the navigation bar.

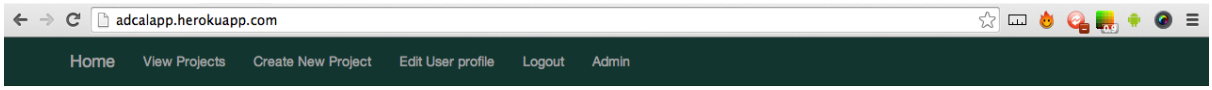


Figure 19 Admin Navigation Tab

When the Admin link is selected the Admin Dashboard is displayed. It lists recently added feedstock's and recently signed up users. The Ad Calculator link will display the app home page.

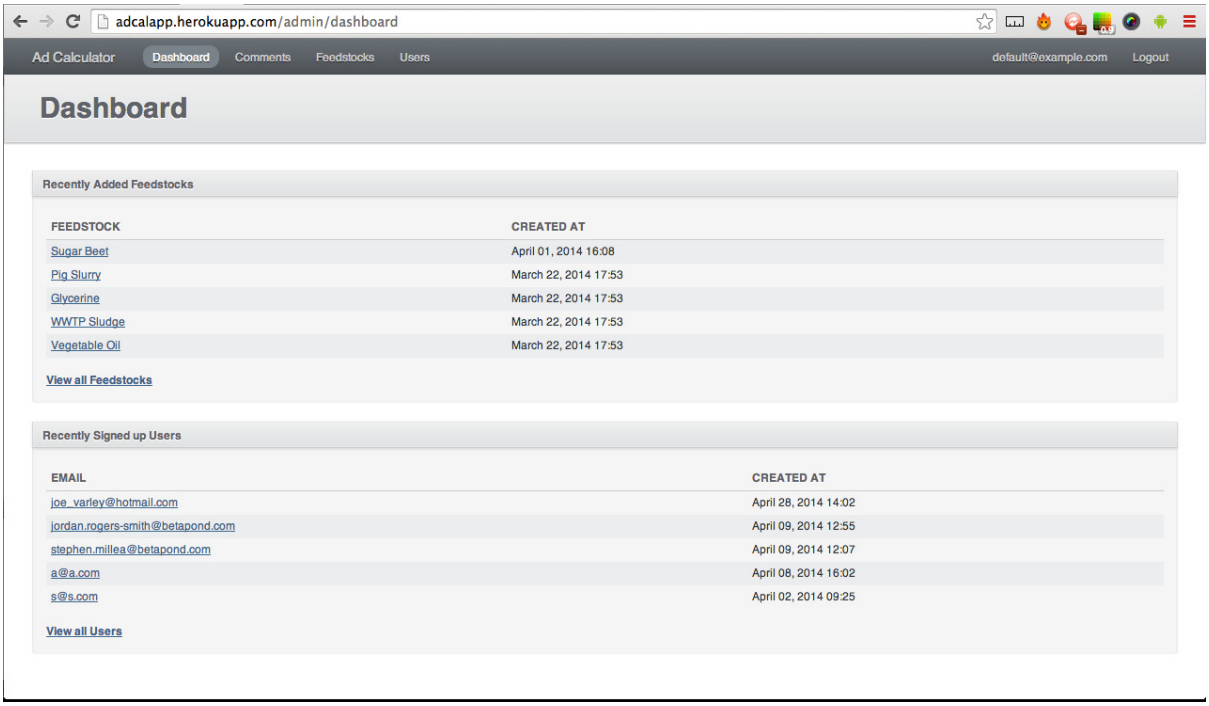


Figure 20 Admin Dashboard Page

The feedstock page displays feedstocks that were saved to the database. Each feedstock can be viewed, edited, or deleted by selecting the appropriate link. A new feedstock is created by pressing the New Feedstock button.

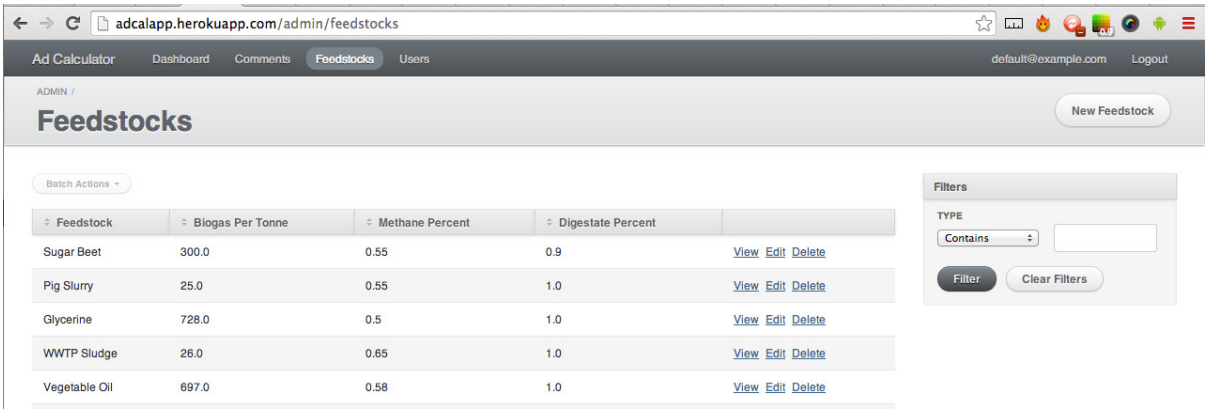


Figure 21 Feedstock List

All values of a feedstock can be edited on the Edit Feedstock page.

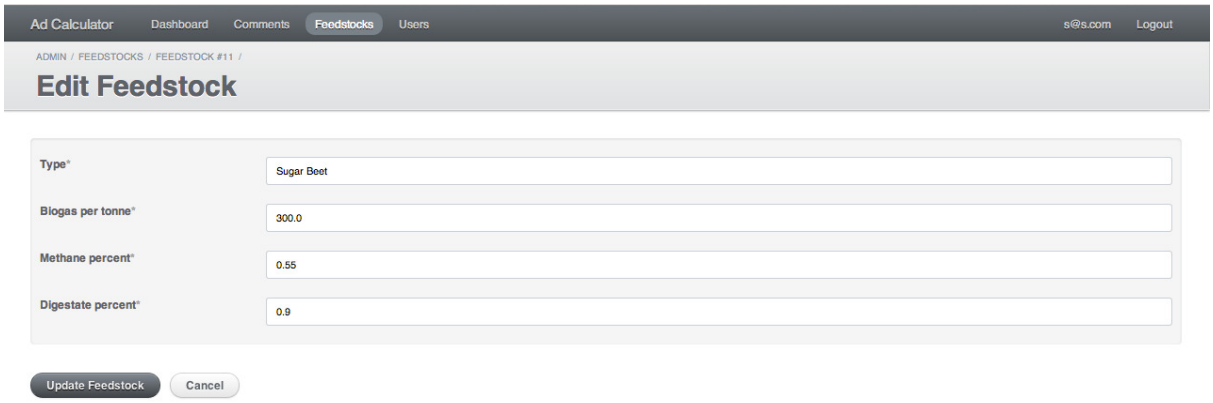


Figure 22 Edit a Feedstock Page

When adding a feedstock ensure that all fields are filled in; type should be a sting, biogas per tonne should be a whole number, and percentages are displayed as decimals i.e. 90% = 0.9. Only an admin user can add a feedstock.

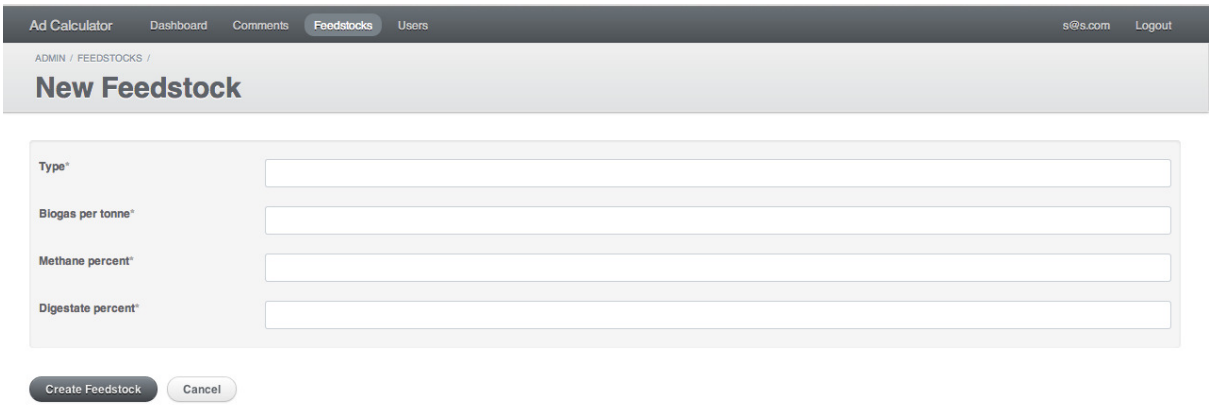


Figure 24 Add a new Feedstock Page

The user page displays all users signed up to the app. User details can be viewed, edited, or deleted by selecting the appropriate link.

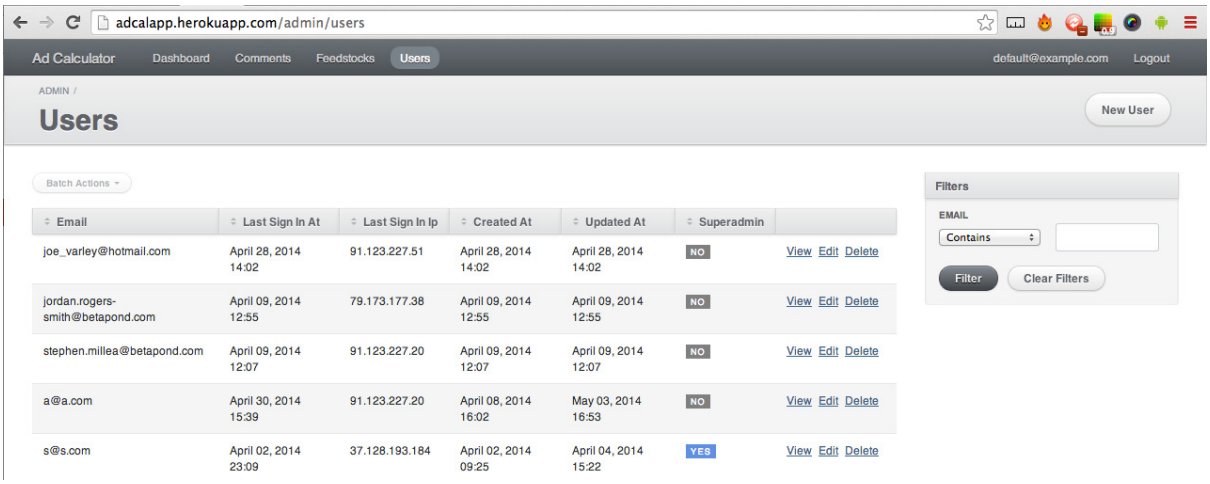


Figure 23 User List Page

When editing a user details admin access can be granted by ticking the Superadmin box.

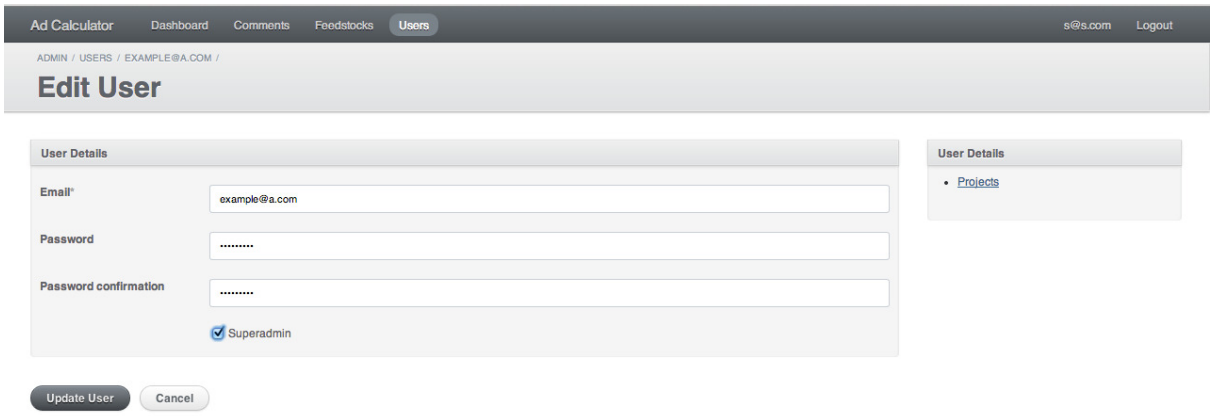


Figure 25 Edit User Page

The admin user can view a user's projects by selecting the projects link.

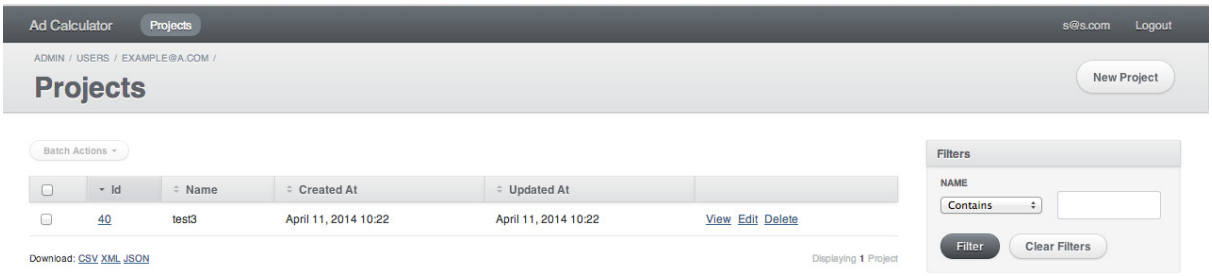


Figure 26 User Project Page

When the view project link is selected a list of calculations belonging to the project are displayed.

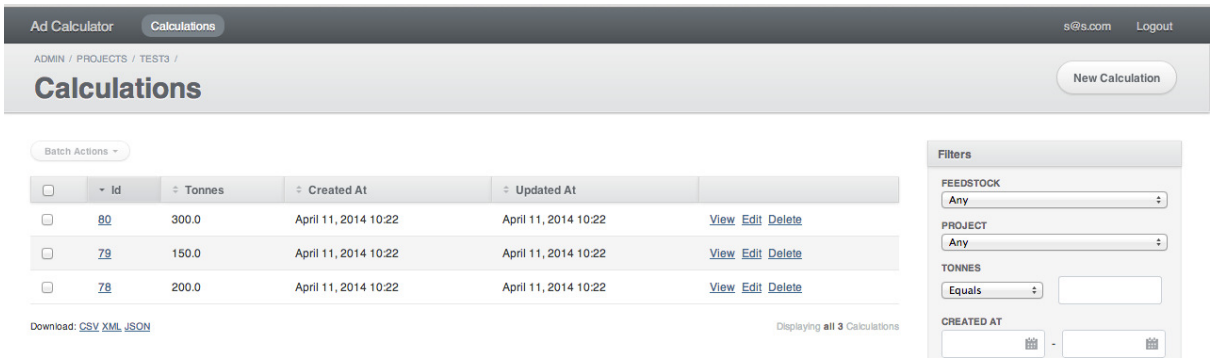


Figure 27 User Project Calculations Page

The details of any calculation can be viewed by selecting the view link.

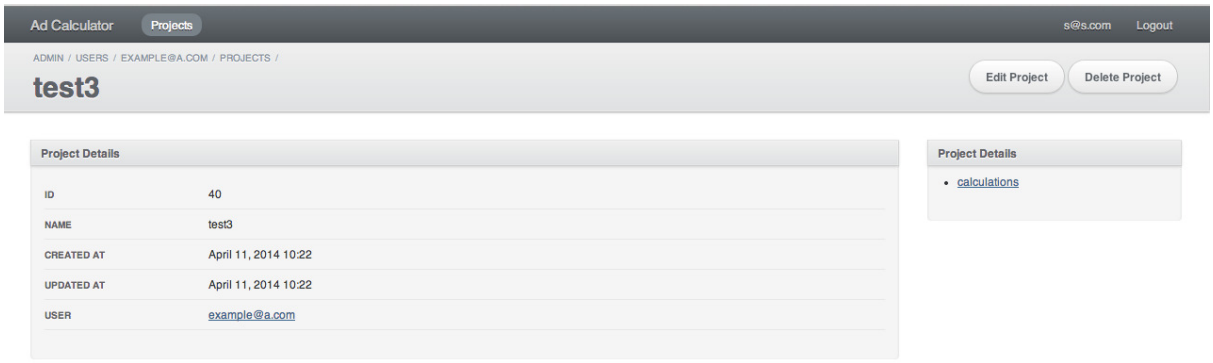


Figure 28 User Project Individual Calculation Page

Chapter 5 Software Design

This chapter describes the software design and implementation. It outlines the framework and hosting service used for the app, provides a description of the model relationships, provides an overall view of MVC relationships, outlines key features, describes how the code was tested, and outlines how the app was validated.

5.1 Introduction

The software was designed to be a calculator app. The options were to design an Android, IOS, or a web app. A web app was chosen because the app will be available on all devices with a web browser; specific CSS styling was implemented for mobile devices using Bootstrap 3. An Android version may be considered for future versions of the app.

The framework options considered for the app was the Play Framework [8] & Ruby on Rails [2]. Ruby on Rails was chosen as it is the framework that is used in Betapond the company where the author is undertaking work placement; support was available when the author ran into problems. The project was used an opportunity to learn Ruby on Rails.

The cloud services considered to host the app were Horoku & Cloudbees. Heroku was chosen because it is compatible with Ruby on Rails and is used extensively by Betapond. Using Heroku required the author to learn how to use GitHub version control.

5.2 High Level Design

The app has four models, Feedstock, Calculation, Project, and User. The models map database tables. A user has many projects, a project has many calculations, and feedstocks have many calculations

The app has three controllers, Application, Visitors, and Projects. The visitor controller renders the home page. The project controller renders the project list page, pages to create and edit a project and all calculation result pages.

The project has three view folders, visitors, projects, and devise. The project folder contains the pages that displays project list, new and edit pages, and calculation result pages. The visitor's folder contains the home page. The Devise folder contains pages related to login and signup and edit user details.

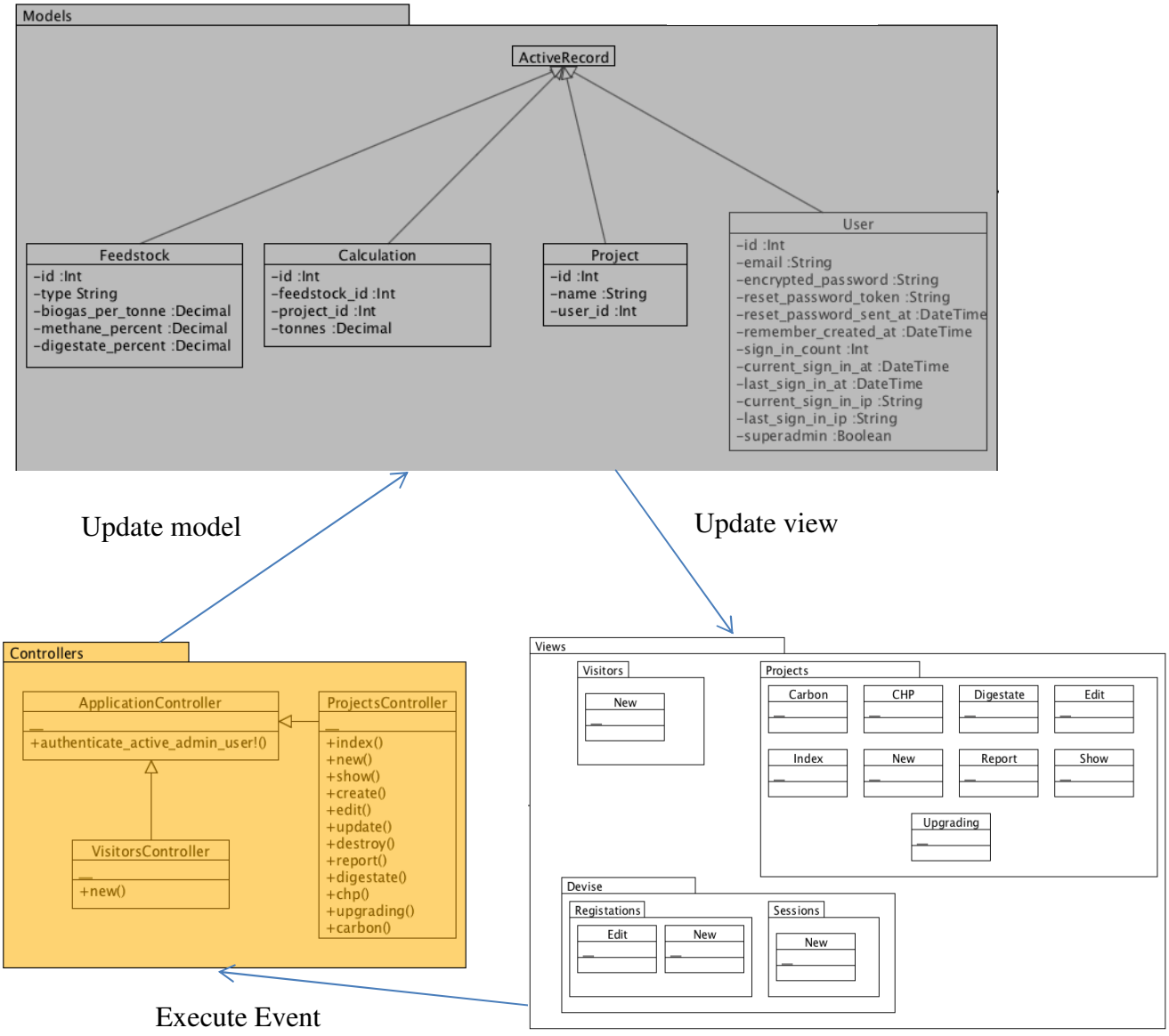


Figure 29 Model, Controller, and Views

5.3 Detailed Design

The calculation model contains methods to calculate biogas and methane output. Conversion methods are used to calculate kilogrammes, gigajoules, and megawatt hours (MWH) of methane. Digestate methods calculate digestate and digestate nutrient volumes.

The project model contains methods that calculate total values. If a project has multiple calculations the total values from each calculation method in the calculation model are added together to get a project total. The total methods values are then used to calculate the potential electricity and heat that could be generated using CHP. A generator size method is used to determine the tariff for electricity a higher tariff is used if the generator is less than 0.5 megawatts. A method is used to set the electricity tariff based on the generator size. Methods then calculate the potential income from heat and electricity generated by CHP.

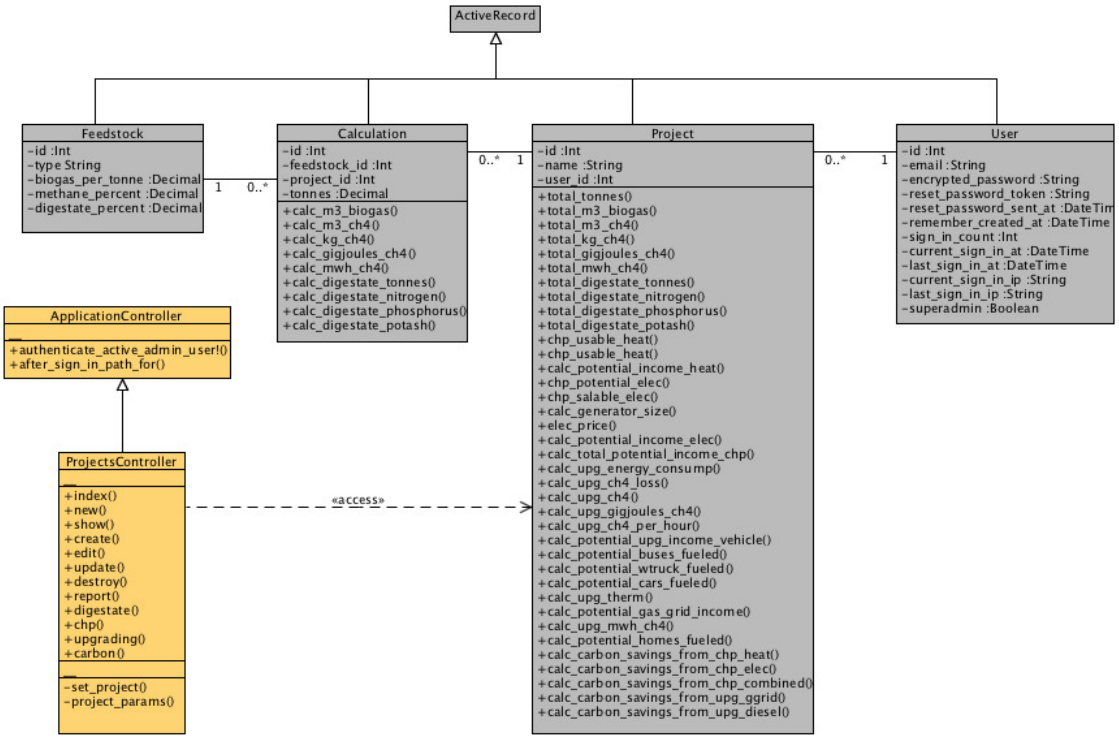


Figure 30 Models and Controllers

Upgrading methods calculate the amount of methane available after upgrading the biogas. An Upgraded methane per hour method is used to determine if upgrading is feasible. Methods calculate the number of busses, waste trucks, private cars, and households that could be fuelled annually with the upgraded biogas. To calculate the potential income for upgraded biogas it is assumed that the biogas is replacing diesel fuel. A method converts the upgraded biogas into therms, this is the unit is which wholesale gas is sold, to enable the potential income from gas grid injection to be calculated. Carbon savings methods calculate the tonnes of carbon dioxide that would be saved by using biogas to generate heat, electricity, as a vehicle fuel, or injected into the gas grid.

The user model was generated with the Devise gem [3]. The devise gem includes a current_user method which is used to only display projects that belong to the current logged in

user. When creating a new project the current_user method is used to identify the user id [3]. A superadmin Boolean field was added to the user model to identify an admin user. If the current user has admin privileges an admin tab appears in the navigation tab.

The project controller contains the default Rails actions index, new, show, create, edit, update, and destroy. Five additional actions report, digestate, chp, upgrading, and carbon were added to display the various calculation pages. The private action set_project is used to find a project id. The private action project_params lists the parameters that can be saved to the project model. To enable a calculation to be saved in a project form, the calculation parameters had to be added to the project_params list.

The application controller contains a method to authenticate an admin user and a method to set the page that is displayed after a successful login.

5.3.1 Design Features

The following are the main design features of the app:

Dynamic Buttons and Navigation Bar

When a user lands on the homepage they have to options to login or signup via the navigation bar or by signup and login buttons. When a user logs in or signs up the buttons change and the navigation bar has additional tabs. The dynamic buttons are achieved by checking if a user has logged into the app, if a user is logged in the buttons relating to a project are displayed, if a user is not logged in the login and signup buttons are displayed. A similar approach is taken with the navigation bar; additional tabs allow a user to edit their details and to logout.

Nested Form

The new projects page contains a nested form which allows a user to save data to multiple models in a single form; in this case the project name is saved to the project model and tonnes and feedstock type are saved to the calculation model. An online tutorial was used as a reference to create this form [9]. To create a nested form; the rails method accepts_nested_attributes_for was added to the project model, this allowed a calculation to be saved through project. A form builder is used to generate calculation fields. In the project controller a single calculation field is generated.

Dynamic Form

When creating a new project the user has the option of adding calculation fields using jQuery. To remove a field, the hidden field _destroy and a remove link were added. The JavaScript code checks if the form is loaded and if any elements have a class of remove_fields. If remove_fields is present the hidden field before the link is set to one. The closest method is used to find the fieldset that wraps all calculation fields and hides it. The preventDefault() method prevents the default link behaviour [9].

To add fields a link is added to the form. An application helper is required to add calculation fields. This method builds an instance of the parent. It takes a parent instance and calls the associated method, which will be the child, call klass on that to get the child class, and call new to create a new child. It grabs the object's object_id to generate a unique value [9].

fields_for is called on the form builder, passing in the association and the child instance just created so that it builds fields for the child and passes a unique id as the child_index. The partial for this association is rendered, in this case _calculation_fields. A link is generated with a class of add_fields and some -data attributes including the unique identifier and all the field data generated [9].

The JavaScript checks for any links with add_fields. When this event is triggered the current time is stored and then generates a regular expression based on the links data-id attribute, which contains the unique identifier created in the helper method. The new fields are inserted before the link by fetching encoded HTML from the data-fields attributes and replacing the unique identifier with the current time. This ensures that each added field will have a unique index based on the current time [9].

Graphs
When a project is created the calculation results are displayed in graphs. The pie charts and column charts are created with the "chartkick" gem [5]. Charts are created dynamically by retrieving calculation values associated with a chosen project from the relevant database tables.

Hide and unhide HTML tables
The user has the option to view calculation results in table format. This is achieved by using the HTML hidden attributes, when the page is displayed the table is hidden. The jQuery show and hide functions were used to display and hide the table [10]. The JavaScript was converted into CoffeeScript using an online converter [11]. The "More Info" and "Hide" buttons use the id attribute to call the relevant JavaScript functions.

Popup Information
Additional information is available by hovering over links. Bootstrap 3 was used to implement the pop up boxes (tooltips) [12]. When a user hovers over the relative link "tooltip" and it is wrapped inside a HTML div with an id of "hint" the pop up message is displayed.

Upgrading Dynamic Message

The value of upgraded methane for chosen project is calculated and depending on the value a different message and background are displayed on the upgrading page. The messages that can be displayed are not feasible, marginal, and feasible.

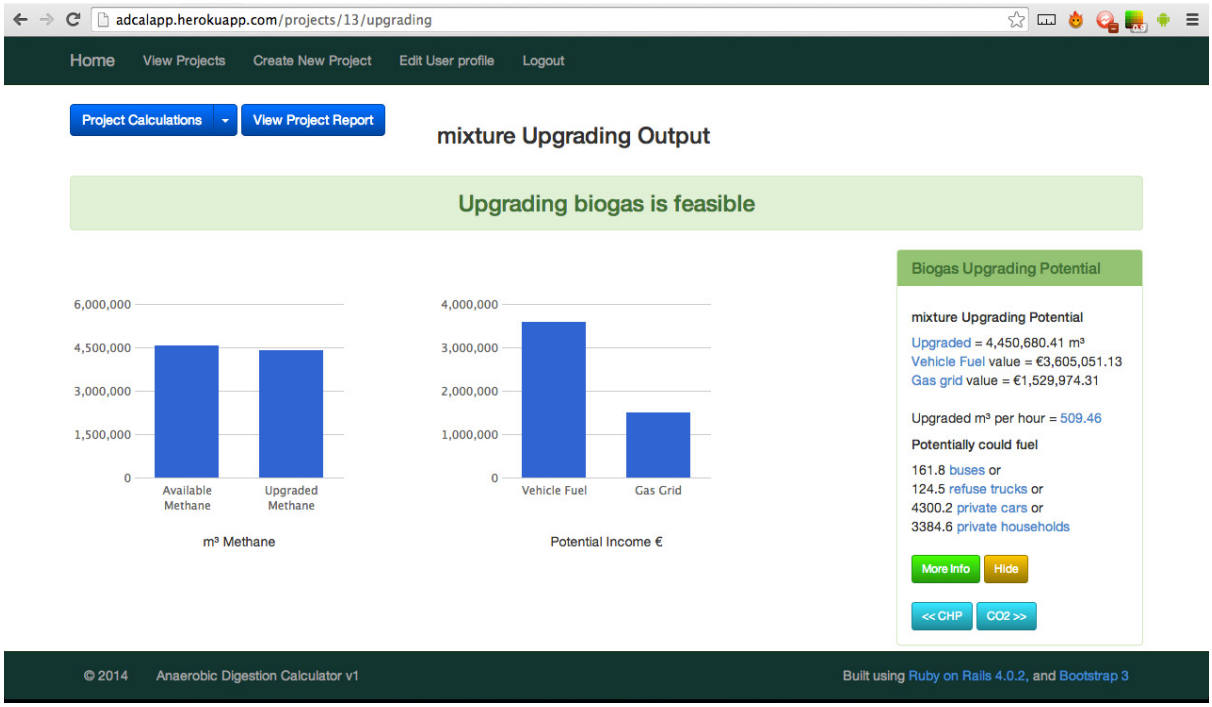


Figure 31 Upgrading Dynamic Message

5.4 Data Storage

There are four database tables in the app user, project, calculation, and feedstock.

The user has a many to one relationship with projects. It stores the user email and encrypted password [3]. A superadmin Boolean is used to determine if the user has can have admin access to the app.

Projects have a many to one relationship with calculations. It stores the project name and the user id; this is used to associate a project to a user.

Calculation has a one to many with relationship with project and feedstock. It stores a tonnes value, the project id and feedstock id.

Feedstock's have a many to one relationship with calculations. It stores the type of feedstock, a biogas per tonne value, a methane percent value and a digestate percent value. This table is preloaded with data, only an admin user can add new feedstock's or alter existing values.

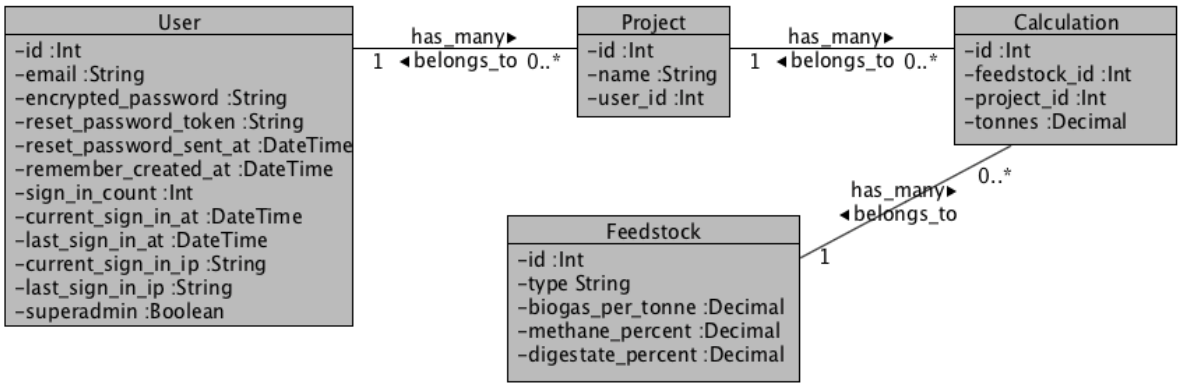


Figure 32 Data Storage

5.5 Verification

RSpec is a behaviour driven test framework for the Ruby language. The rspec-rails and factory_girl_rails gems were installed to enable RSpec testing. The rspec-rails adds RSpec with some additional Rails features. The factory_girl gem is use to set up test data with factories [7].

Factory Girl is used create a project factory. It is then called in the RSpec test using the build parameter. This test tests the name validation; if no name is present and there are no errors the test fails.

To test calculation methods a new project is created, the relevant values are passed into the calculation, and to pass the value calculated by the method must equal the expected value. Tests were written for all model methods written by the author.

The green text indicates that all the tests have passed.

```
Feedstock
  is is invalid with a duplicate type
  is invalid without a type
  is invalid without a methane_percent
  should have many calculations
  has a valid factory
  returns a type as a strings
  is invalid without a digestate_percent
  is invalid without a biogas_per_tonne

User
  should have many projects

Project
  calculates correct upgrading energy consumption
  calculates correct income potential  CHP heat
  calculates correct upgraded methaner per hour
  calculates correct co2 savings from CHP heat
  should have many calculations
  calculates correct co2 savings from gas grid
  calculates correct income potential CHP electricity
  calculates correct potential buses fueled
  calculates correct potential waste trucks fueled
  calculates correct CHP usable electricity
  calculates correct upgrading potential vehicle value
  should belong to user
  calculates correct potential gas grid value
  calculates correct upgraded methane volume
  calculates correct potential mwh upgraded
  calculates correct upgrading methane loss
  calculates correct co2 savings from vehicle diesel
  is invalid without a name
  calculates correct potential cars fueled
  has a valid factory
  calculates correct CHP generator size
  calculates correct upgraded gigajoules
  calculates correct total potential income from CHP
  calculates correct co2 savings from CHP electriciy
  calculates correct upgraded therms
  calculates correct CHP usable heat
  calculates correct co2 savings CHP combined
  calculates correct potential homes fueled
  calculates correct CHP salable Electriciy
  calculates correct CHP salable heat

Calculation
  calculates correct methane kg value
  has a valid factory
  calculates correct phosphorus volume
  should belong to project
  calculates correct methane value
  calculates correct potash volume
  calculates correct digestate volume
  is invalid without a tonnes
  calculates correct methane gigajoules value
  should belong to feedstock
  calculates correct methane mwh value
  calculates correct nitrogen volume
  calculates correct biogas value
  is invalid without a feedstock_id

Finished in 0.24443 seconds
53 examples, 0 failures

Randomized with seed 9383
```

Figure 33 Model tests

5.6 Validation

When extra functionality was added to the app it was manually tested. After the final iteration the URL of the app was shared with work colleagues and friends they were encouraged to use the app and find bugs. After validation of the final iteration the following improvements were made: spelling mistakes on the home page were corrected, clearer instructions were displayed on the new project page, and feedstocks were listed in alphabetical order in the dropdown list.

Chapter 6 Discussion and Conclusion

This chapter contains a project review and conclusion. It reviews the project solution, outlines what would be done differently if starting again, identity's major problems encountered and how they were handled, provides a list of the key skills that were learned during the project, lists proposals for future work and summarises the project and the author's solution.

6.1 Project Review

The author successfully created a Ruby on Rails app that calculates energy output from anaerobic digestion. All user stories were implemented. Bootstrap 3 was used for styling; the buttons [13] and navigation bar [14] CSS styles were overwritten to avoid a generic looking bootstrap website.

When creating the app the author tried to recreate the Excel worksheets in the app. If the author was starting again the Excel model would only be used as a guide only. The author required guidance from their project supervisor to come up with a suitable model structure. The author's workplace mentor was consulted to ensure that the proposed structure was achievable in a Rails project.

The author had to get advice from their workplace mentor on how to write RSpec test for calculation methods. The methods had to be rewritten; values were passed into the methods, to enable tests to be written. The author required help from a work colleague to display calculation results in a pie chart. The pie charts required data to be in an array of hashes.

6.2 Key Skills

The following are new skills that the author learned when working on the app:

1. Create a Ruby on Rails project, run migrations to alter database tables, use Ruby gems to add additional functionality, and check log files if something unexpected happens.
2. Write RSpec tests for a Rails app and use Factory Girl to create test data.
3. Use GitHub version control, create a local repository, push to a remote repository, commit local changes, create branches, merge branches, and delete branches.
4. Deploy project to Heroku cloud service, push code using GitHub, run migrations, and check Heroku logs when app crashes.

6.3 Future Work

The following are project enhancements that would improve future versions of the app:

Include tariffs from the UK and other countries. When a user is creating a project they will have the option to choose a country. The tariffs will be used in the potential income methods. This will allow a user to compare the tariffs in different countries; this is especially relevant in Ireland where Northern Ireland has different tariffs to the Republic of Ireland.

Modify the methods relating to potential income; replace the hardcoded values with dynamic values that could be changed by a user. This will make the app more interactive, at the moment if a tariff changes there is no way for a user to update the value.

Develop an android version of the app. The app can be displayed on mobile devices however the graphs are not suited to small screens. Developing an android version would improve the user experience on android devices.

6.4 Conclusion

The objectives of the project were to develop a user friendly web app that calculates the energy output from organic materials through AD. AD is a process where biogas is produced from organic materials in an oxygen free environment. The end-uses for the biogas include generation heat, generating electricity, use as a vehicle fuel, and injection into the gas grid. The algorithms to calculate the potential income from the various end-uses are not freely available to the general public. The project was broken down into user stories to define the features required. The project was completed in four iterations; the user stories were assigned to an iteration. Ruby on rails was chosen as the framework to build the app; the project was used as an opportunity to learn a new framework. The app is deployed on the Heroku cloud service.

A user can login and signup to the app, can create a new project or view existing projects, view project calculation data on separate pages, and view a project report. A user with admin privileges can add or edit feedstock values, grant admin access to other users, and monitor user activity. The app has four models, user, project, calculation, and feedstock. A user has many projects, a project has many calculations, and a feedstock has many calculations. When a project is saved the app saves the data to the relevant database tables. This data is retrieved and run through methods in the calculation and project models to display graphs on the various calculation pages. RSpec behaviour driven testing was used to test the methods. Features that will be considered for future version include, including tariffs from other countries, replace hardcoded values relating to potential income methods with dynamic values, and create an android version of the app.

Bibliography

[1] Biowatts, “biowatts.org,” 2013. [Online]. Available: http://www.biowattsonline.com/. [Accessed 04 March 2014].

[2] rubyonrails.org, “RailsGuides,” 2013. [Online]. Available: http://guides.rubyonrails.org/getting_started.html. [Accessed 13 December 2013].

[3] Plataformatec, “Devise,” 2014. [Online]. Available: http://devise.plataformatec.com.br/. [Accessed 25 March 2014].

[4] ActiveAdmin, “The administration framework for Ruby on Rails,” 2014. [Online]. Available: http://www.activeadmin.info/. [Accessed 25 March 2014].

[5] ChartKick, “ankane/chartkick,” 2014. [Online]. Available: https://github.com/ankane/chartkick. [Accessed 24 April 2014].

[6] jQuery, “About jQuery,” 2014. [Online]. Available: http://learn.jquery.com/ . [Accessed 10 April 2014].

[7] A. Sumner, eyeryday Rails Testing with RSpec, Leanpub, 2013.

[8] Play, “The main concepts,” 2013. [Online]. Available: http://www.playframework.com/documentation/1.2.7/main#mvc. [Accessed 12 December 2013].

[9] R. Bates, “Nested Model Form (Revised),” 2012. [Online]. Available: http://railscasts.com/episodes/196-nested-model-form-revised?. [Accessed 15 March 2014].

[10] K. Swedberg, “LeaningjQuery,” 2006. [Online]. Available: http://www.learningjquery.com/2006/09/basic-show-and-hide/. [Accessed 02 April 2014].

[11] Js2Coffee, “JavaScript to CoffeeScript compiler,” 2014. [Online]. Available: http://js2coffee.org/. [Accessed 2 April 2014].

[12] Bootstrap, “JavaScrip,” 2014. [Online]. Available: http://getbootstrap.com/javascript/#tooltips. [Accessed 5 April 2014].

[13] C. Park, “Beautiful Buttons for Twitter Bootstrappers,” 2014. [Online]. Available: http://charliepark.org/bootstrap_buttons/. [Accessed 05 April 2014].

[14] S. Marchal, “Generate your own Bootstrap navbar,” 2014. [Online]. Available: http://smarchal.com/twbscolor/?bd=113631&bh=102d10&cd=a3a3a4&ch=fffcfe.

[Accessed 07 April 2014].

[15] CloudBees, “How It Works,” 2013. [Online]. Available: http://www.cloudbees.com/platform/how-it-works.cb. [Accessed 12 December 2013].

[16] Heroku, “How Heroku Works,” 2013. [Online]. Available: https://devcenter.heroku.com/articles/how-heroku-works. [Accessed 13 December 2013].

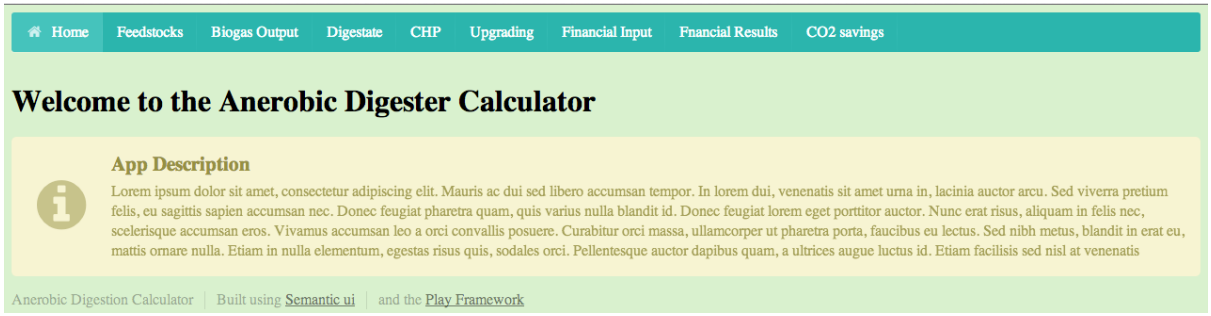
[17] RailsGuides, “Rails Routing from the Outside In,” 2014. [Online]. Available: http://guides.rubyonrails.org/routing.html. [Accessed 6 April 2014].

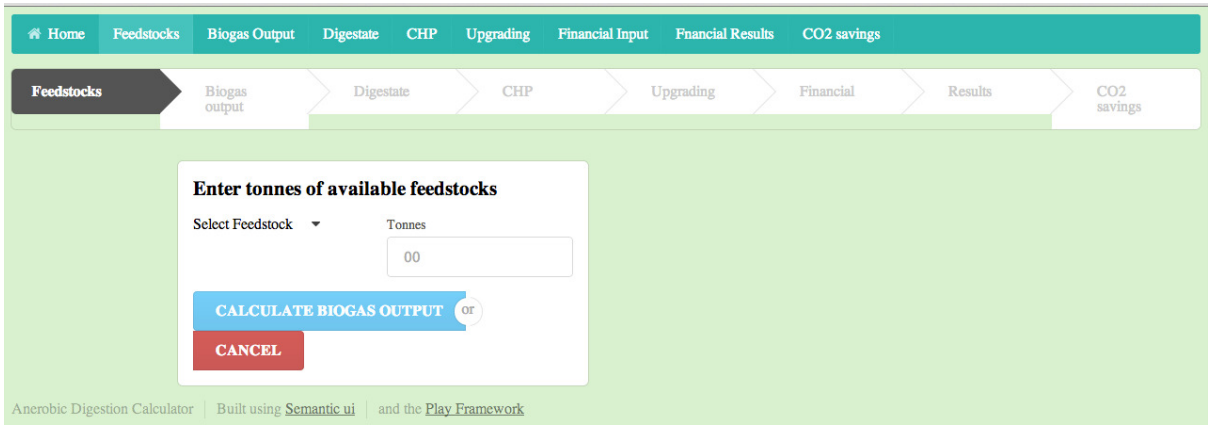
[18] Git, “Pro Git Book,” 2009. [Online]. Available: http://git-scm.com/book. [Accessed 25 April 2014].

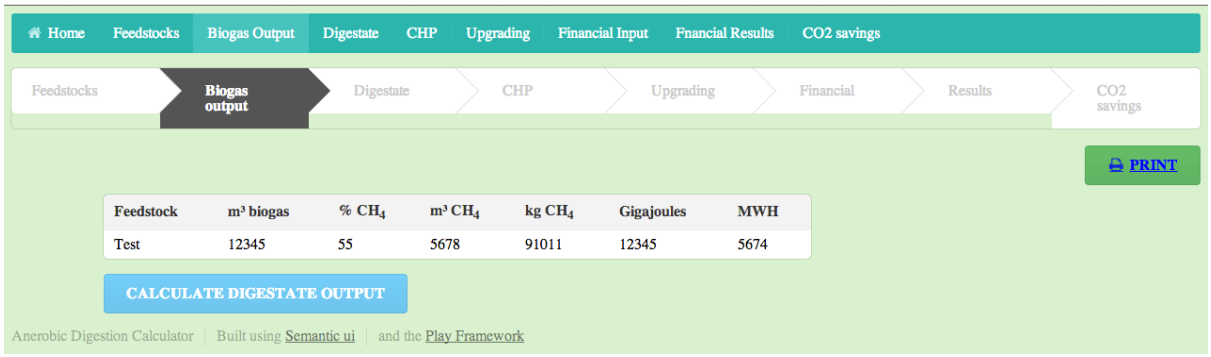
Appendices

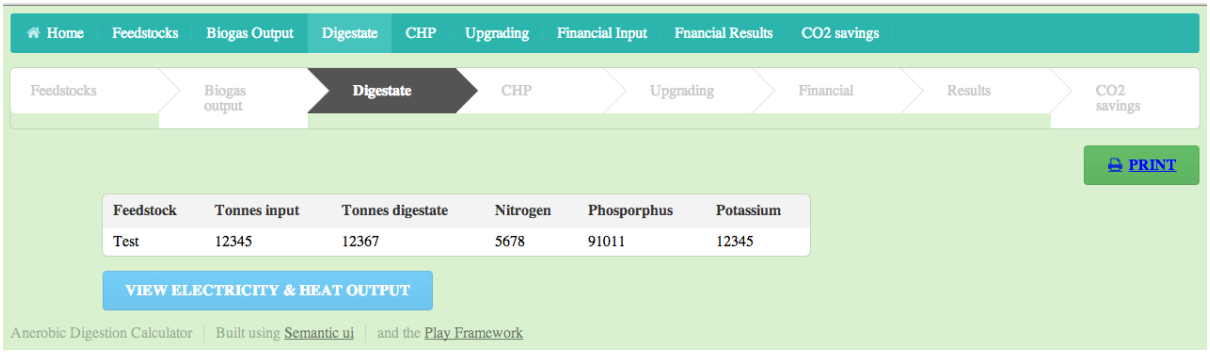
Appendix A

A1 Static Website









Appendix B

B1 Java Models and JUnit Tests