



Technical Report

MAGILE - A Project Portfolio Management Reporting Tool

Prepared by: Claire McGuckin, Higher Diploma in Science Cloud Computing

13 September 2015

Student number: x14127695

TECHNICAL REPORT	1
EXECUTIVE SUMMARY	3
1. INTRODUCTION	4
1.1 BACKGROUND	4
1.1 AIMS	4
1.3. TECHNOLOGIES	5
1.3. STRUCTURE	6
2. SYSTEM	7
2.1 REQUIREMENTS	7
2.2 DESIGN AND ARCHITECTURE	9
2.4 IMPLEMENTATION	10
2.4 TESTING	10
2.5 GRAPHICAL USER INTERFACE (GUI) LAYOUT	13
2.6 CUSTOMER TESTING	16
3. CONCLUSIONS	17
4. FURTHER DEVELOPMENT OR RESEARCH	19
5. REFERENCES	20
6. APPENDIX	21

Executive Summary

With the vision of empowering management to make the right decisions effectively, this document outlines the technical details associated with the specification, design, implementation of Magile, a system designed to give a realtime view of projects and products across the portfolio at The Globe.

With recent efforts to initiate the rollout of agile methodology across the department, the project management office at The Sun are experiencing conflicts in the asks from the senior management team, the organisational leadership team (external to the department) and the agile way of working.

The team are continuously answering to senior management demands for reporting and metrics on the projects ongoing within the department. The team are experiencing pushback from project managers (and scrum masters/product owners), who are working to instil the agile way of thinking across the department, supporting and promoting a light-touch model. Light reporting, light documentation, and verbal project updates through mediums such as the teams daily stand-ups.

However, regardless of the approach to project management, senior management quite obviously cannot attend all of these daily meetings, backlog grooming sessions and retrospectives, yet still need continual visibility into what is going on within the projects. Not only to ensure they are generating the maximum return on investment, and making the right portfolio decisions, but because they have questions to answer from their managers, and to them agile often means nothing other than being able to move quicker and easier - yet PMO reporting is taking longer than ever!

Through the use of a system developed in ruby on rails, to be integrated with the project management system, JIRA, Magile was designed and provide realtime reporting and metrics for to give senior management, to give the visibility they need, to make the right decisions at the right time.

1. Introduction

1.1 Background

Globe Inc. sell products to governments around the world. Currently employing over 65,000, their products range from health services to defence products. One department, The Sun, specialise in analysing the data associated with the worlds population so that the other departments within the organisation can use this data to facilitate discussions and grow the company's revenue through selling their products to governments worldwide. Additionally, Globe Inc., sell this data through a suite of analytical products to governments and interested parties globally.

The Sun have recently purchased a tool called JIRA, developed by Atlassian, Inc., to aid the development of these analytical software products through platforms and associated projects. Although the system is very effective in achieving its goal of planning and tracking the work of project teams, the only visibility senior management have into the work of their project teams is through the weekly report about the health and status of projects, submitted by project managers. Therefore, senior management are depending on the proficiency and honesty of their staff, and can see little or no metrics to back up the words.

The department have purchased a subscription of 100 licenses, all of which are currently being used by the project teams. If senior management want to login to the system to view the tasks being done by their project teams, they will need a license and therefore the department will need to upgrade to 500 subscriptions per month. Given the immature state of the integration of the tool within the department, senior management are hesitant to make additional investment in the tool. Additionally, senior management want to see a portfolio view of platforms, projects and teams as well as the individual project views, something which out of the box dashboards provided by the tool do not facilitate. Furthermore, there are other tools being used and support activities being carried out by portfolio management team in attempt to understand the portfolio and respond to senior management demands. Other processes include Stage Gate, where ideas and project requests directed to department are analysed to understand the needs, demands and ROI potential before being executed. Therefore, the department need a system that will give senior management a snapshot view of the platforms, projects and teams, and allow them to view a more granular level of detail against each of these, as and when required.

1.1 Aims

The aim of this project is to alleviate the pain points experienced by the project management team, and to give senior management an optimal view of the project portfolio, as well as capturing innovative ideas (from employees throughout the company) and getting a view of the customer satisfaction associated with the department deliverables.

The scope of this project is to develop a system to integrate data from existing systems being used throughout the department, and allow the necessary users to add information needed by management, currently being

communicated by email. The user interface should provide a positive user experience, and with the meeting the requirements of the marketing department while getting the right message to the right people.

Having worked in portfolio management across multiple industries for a number of years, these requirements have been well thought out, and ensure the smallest tasks, typically taking a high percentages of resource time or delivered with minimal accuracy have been addressed through this project, or will be addressed through the evolving product. Ultimately, these deliverables are to ensure management have the right information to make the right decisions at the right time.

1.3. Technologies

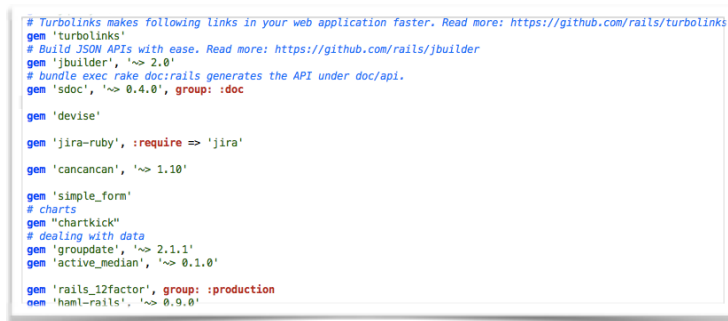
The networks, servers and storage associated with the application will be hosted on Heroku, Heroku is a cloud platform as a service, initially developed for the ruby programming language. The platform facilitated the launch of the application without infrastructure costs, and allows the application can be run, managed and supported as it develops in size and complexity.

The application is developed through the open-source web framework, ruby on rails, allowing basic functions, such as generating html through the views, processing forms for employees, project requests, or accessing the Amazon RDS database. Given the applications that have been built through rails, such as Twitter, Air BNB and Bloomberg, there is a large pool of open source libraries that can be used to develop the application, and ensure optimal usage of the rails framework. The application is enhanced through the use of ruby gems, for example the devise to control access to the application.

The web browser uses HTML, CSS and JavaScript to display the webpages contained in rails. HTML is used for creating documents with structure, a combining text and images.

CSS is used to apply an appearance to the text and image content, and JavaScript will be used for the interactive features, such as the expanding menus on the homepage. Rails creates these files dynamically using Ruby, to create a web application. This facilitates the display of database items through the webpages, for example resource names and related data.

The application is using PostgreSQL as the database storing information being retrieved by the application, such as project ideas and customer feedback.



```
# Turbolinks makes following links in your web application faster. Read more: https://github.com/rails/turbolinks
gem 'turbolinks'
# Build JSON APIs with ease. Read more: https://github.com/rails/jbuilder
gem 'jbuilder', '~> 2.0'
# bundle exec rake doc:rails generates the API under doc/api.
gem 'sdoc', '~> 0.4.0', group: :doc

gem 'devise'

gem 'jira-ruby', :require => 'jira'

gem 'cancancan', '~> 1.10'

gem 'simple_form'
# charts
gem "chartkick"
# dealing with data
gem 'groupdate', '~> 2.1.1'
gem 'active_median', '~> 0.1.0'

gem 'rails_12factor', group: :production
gem 'haml-rails', '~> 0.9.0'
```

Fig 1: Gems used for the application development



Fig 2: the use of JavaScript for the interactive design on the system homepage.

1.3. Structure

The sections throughout this document outline the system, and associated technologies in more details, in the following structure:

Section 2: System

This section outlines the system design and development, to include the user requirements, the design and system architecture as well as components used, the implementation including classes and functions, quality assurance testing, user acceptance testing, the GUI (Graphical User Interface) layout and related marketing requirements, and a system evaluation.

Section 3: Conclusions

The conclusions outline the advantages and disadvantages associated with the system, as well and the opportunities and limitations.

Section 4: Further Development or Research

This section outlines the features that could be development with more resources, and the commercial potential of the application.

Section 5: References

Various references have been used to design and develop the idea and application. These are referenced in Section 5.

6. Appendix

The Appendix includes all other material used in the application development process, to facilitate and enhance the application development and deployment.

2. System

2.1 Requirements

Problems described in earlier versions of this document were identified before project initiation, and then reviewed in more detail before the documentation of the user requirements, formalised in the requirements specification document (Appendix). However, as the implementation of agile evolved throughout the department, these requirements have also evolved, as well as limiting factors such as legal and security approval.

The requirements specification was based on the assumption that *data from systems being used by the company to facilitate communications between teams (such as JIRA) can be extracted and integrated with a web based system*. Because the organisation's JIRA instance is currently hosted in a HP data centre, it does not facilitate the communication of data to other systems. Therefore, the data associated with projects and issues is dummy data from JIRA, and the realtime integration will be completed once the company have migrated to a cloud instance of JIRA, being delivered through a separate project, delayed due to legal sign off, but scheduled to be completed next month. Therefore the realtime integration of JIRA data with this application will be delivered through phase 2 of this project.

2.1.1. Functional Requirements

The original requirements documentation covered five functional requirements, which remain unchanged. These include:

1. Login to the system: the user should be able to login to the system. If the user does not have the required details, they must be able to sign up, or take an option related to a forgotten password.
2. Select appropriate option from landing page menu: the user should be able to select what they want to view or do when they login to the system. Options include platforms, projects, submitting an idea, or customer feedback on a particular project delivery.
3. Submit an idea or project request: the user should be able to submit an idea, or a particular project request through filling out a form with the details to be assessed by the stage gate team before the request is progressed for further analysis and execution.
4. User updates the project summary: fields that are not updated through JIRA can be updated here by project managers, to be used for monthly reporting and meetings related to project governance.
5. Customer submits feedback: the user navigates to the submit feedback option, and updates the form to give feedback about project deliveries.

The views were not included in the original requirements document, but include the ability to view the portfolio summary - the projects per platform, scrum master or executive sponsor, and also a summary view of the individual projects, with the number of project related tasks completed by month, issues currently active and issues by employee.

2.1.2 Data requirements

As mentioned, the data from the JIRA application is dummy data and stored in the application database. Once the migration and integration with JIRA is complete, the data will be retrieved through HTTP requests.

Other database tables include those for the platform and project metadata, to include storing the metadata associated, as well as the project status comments updated biweekly by project managers, customer feedback forms updated as project deliverables are received, and ideas, which will grow both as more requests are submitted, and the organisation grows in size and complexity.

Other growth considerations should be the employee details, which will be used for resource capacity planning functionality to be released in later stages of the project. However, given the size and complexity of the data to be stored in the system databases, the cost of this growth will be limited.

2.1.3 User requirements

The users can login to the system through the popular browsers used in the organisation. These include internet explorer, chrome and firefox. All views are assessable, and users should be able to submit the forms related to projects, ideas and feedback, described in detail above.

These views and forms should be assessable through a wi-fi connection, with considerable bandwidth given the number of charts and associated data being transmitted when pages are refreshed.

There is a growing number of windows surface devices being used in the organisation, and many of the senior management team will be viewing this information on the go, by mobile. The various forms of viewing the application are catered for through the responsive design.

2.1.4 Environmental requirements

A reliable wi-fi connection should be able to run the system effectively and allow successful transmission of all data points. Return times should be minimal.

2.1.5 Usability requirements

Given the (senior) level of the users, and many with limited technical knowledge, the system must be intuitive, and easy to navigate around. The help pages with frequently asked questions, as well as the search functionality should ease the usability of the system.

2.2 Design and Architecture

The application is designed as browser-based user application, and therefore, supported by cloud infrastructure the system can be efficiently scaled to meet the growing demands of the organisation. Limited resource constraints will promote the functionality enhancements and system evolution.

This section describes the design, system architecture and components used. Firstly, the underlying architecture uses the MVC framework, used for all rails applications. This includes:

- **Model** - the Data Layer. This is where the application state is maintained, as well as the data to be used in response to queries. The view is notified and updated with changes.
- **View** - The Presentation Layer, where the models are rendered, and user gestures are sent to the controller to define the behaviour. The controller selects the appropriate view to be used.
- **Controller** - The Application Layer, including the logic used by the application. There is a controller for each of the functionalities outlined above. These controllers map the users' actions to the model, and select which views should be returned to the user.

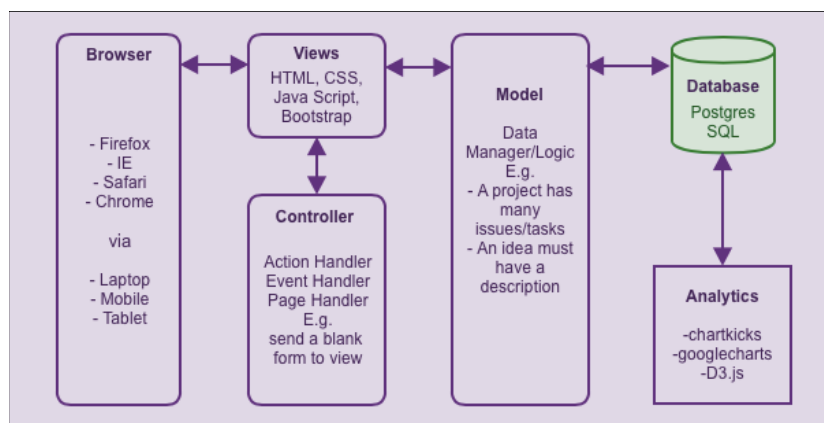


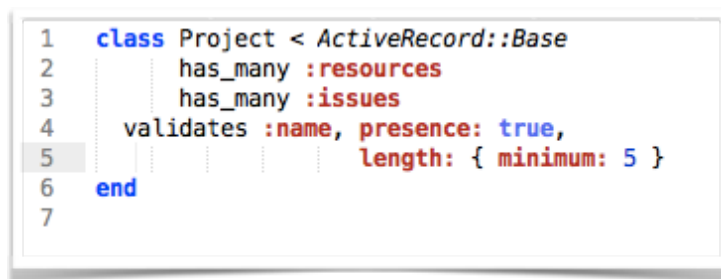
Fig 3: Magile System Architecture

The controller includes the main algorithms logic, defining the rules to be followed by the application in response to user actions. For example, the code below shows that when a user want to create a new idea, the controller sends a blank form to the user, once the idea has been created, the completed form is returned to the user, in the way the projects

```
18   def create
19     @project = Project.new(project_params)
20     if @project.save
21       redirect_to @project
22     else
23       render 'new'
24     end
25   end
```

Fig 4: Controller - create new project code

Database tables include projects, issues, resources and users. The user database is used by the devise gem, used to implement authentication on the application, allowing multiple users to be signed in at once but encrypting and storing a password in the data to validate the user sign in (through POST requests). Project tables are associated with issues and resources. A project can have many resources, and a project can have many issues, as shown in Fig. 5 below, where the application logic is storing this association.



```
1 class Project < ActiveRecord::Base
2   has_many :resources
3   has_many :issues
4   validates :name, presence: true,
5               length: { minimum: 5 }
6 end
7
```

Fig 5: Project Model - Application logic, associating projects with resources and issues

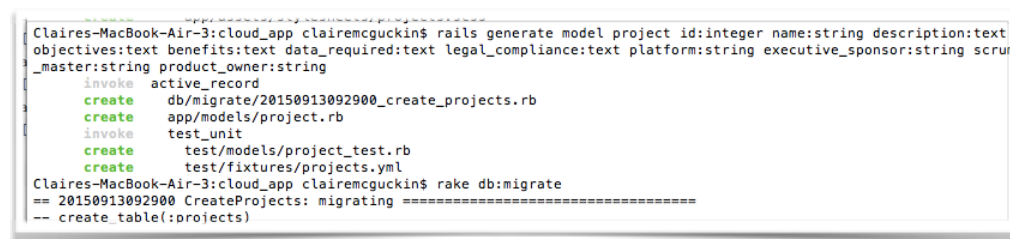
Via the controller, issues associated with projects can be rendered in the application views. Where the user can see the views under a particular project, or add issues to a project.

2.4 Implementation

Describe the main classes/functions used in the code. Consider to show and explain interesting code snippets where appropriate.

The main functions used in the code are included below. These include.

Many rails functions were used to implement the application, to include scaffold, generating models, generating controller, database migrations, but also through external gems (briefly mentioned in the introduction).



```
Claire-MacBook-Air-3:cloud_app clairemcguckin$ rails generate model project id:integer name:string description:text
objectives:text benefits:text data_required:text legal_compliance:text platform:string executive_sponsor:string scrum
_master:string product_owner:string
  invoke  active_record
  create  db/migrate/20150913092900_create_projects.rb
  create  app/models/project.rb
  invoke  test_unit
  create  test/models/project_test.rb
  create  test/fixtures/projects.yml
Claire-MacBook-Air-3:cloud_app clairemcguckin$ rake db:migrate
== 20150913092900 CreateProjects: migrating =====
-- create_table(:projects)
```

Fig 6: To create table model and perform a database migration

Once development was complete the application was deployed to Heroku, firstly by downloading toolbely and then configuring git to create a local repository storing files related to the application. Following this a template was created on Heroku, poster production database was installed and configured.

2.4 Testing

For the purposes of quality assurance, I carried out tests against each of the functionalities described earlier in the document in the development environment before deploying to Heroku, and again once the application was deployed.

The quality assurance went through multiple iterations before all successfully passed, and the application met the requirements as planned. The helped focus on meeting the end deliverables and customer requirements for the project throughout. The week on week results are displayed on the graph below, with the final test results and comments details in Fig. 8

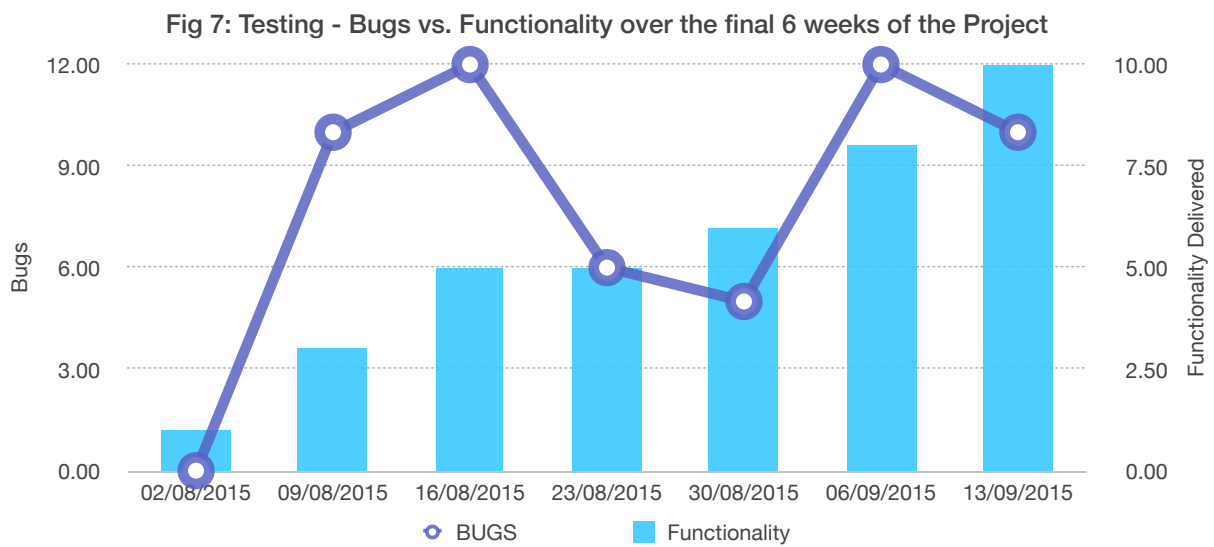


Fig 7: Functionality developed and tested over the final 6 weeks of the project.

Test Case/Acceptance	Pass/Fail	Comments
The user can sign up to the system	Pass	Can sign up but the approval should ideally not be automated - given the potential sensitive data in the system
The user can login to the system	Pass	Can successfully login
The user can request a password reset	Pass	Can reset my password on the system. Should ensure emails do not contain passwords for security reasons
The user is directed to the landing page upon successful login	Pass	Yes - the options on the landing page are all available. These should potentially be customised to the user and their level of access
User can select each of the application options on the landing page	Pass	All can be selected - please see note above about potential enhancement
When the user selects the request menu they can complete the form and submit the request	Pass	Form is submitted and information returned. Enhancement: receive this information via email.
The customer can navigate to the feedback section and complete all fields on the form to give their feedback about the delivery	Pass	Can successfully complete form.
The project manager can update the latest status of the project	Pass	Can successfully update - put shouldn't be able to update the entire project. Reported as bug.
A user can view the latest summary of all projects	Pass	Can view - with dummy data. This should be viewed in realtime with the integration of JIRA data.

Fig 8: Test plan and final results.

Once the system was deployed, additional test were carried out by individuals representing the different system user groups, who were satisfied with the results, and agreed with the enhancements to be included in the next phase of the project. These enhancements are detailed below.

2.5 Graphical User Interface (GUI) Layout

Fig 9. shows the system login page. Here the user choose from the options of:

- **Login:** if they have and know their email and password
- **Sign up:** complete the registration details to sign up for an account
- **Forgot password:** the user can reset their password via this link

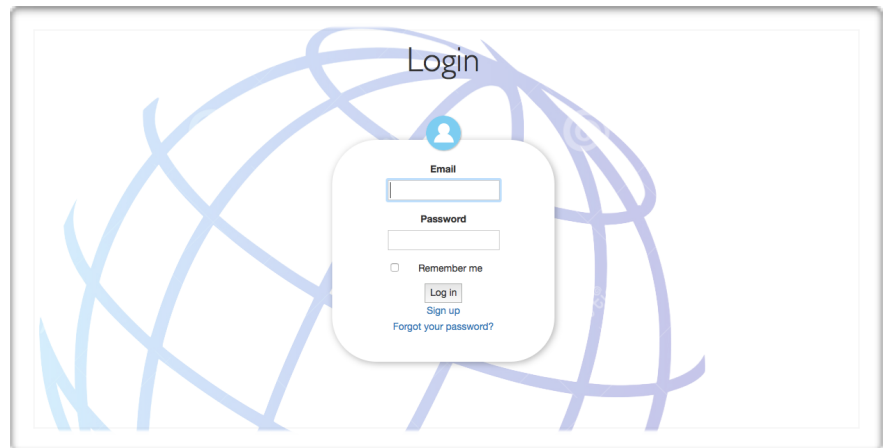


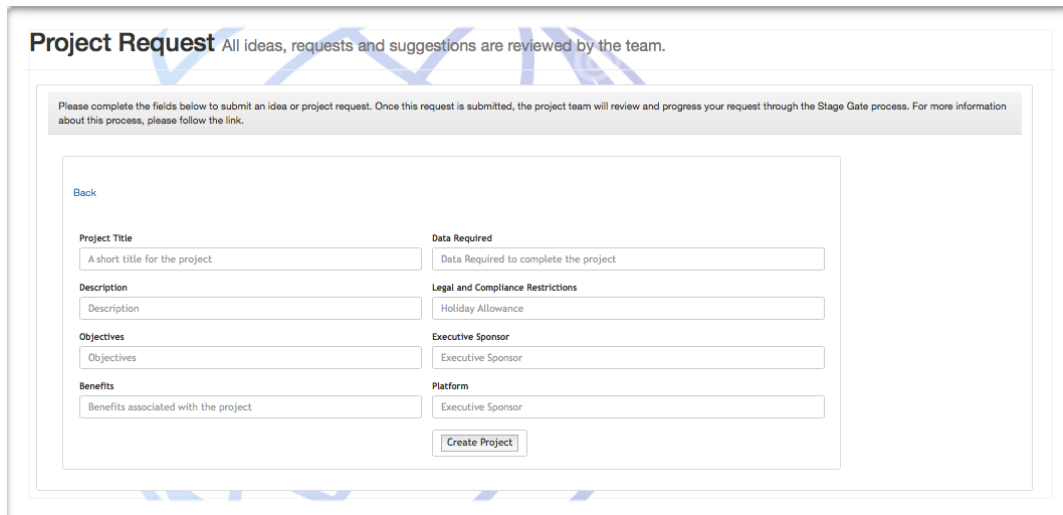
Fig 9: Login page

Fig 10. below shows the landing page the user can see once they are successfully logged in. The options on the landing page are as follows:

- **User:** to edit their user profile or logout
- **Projects:** to view a summary of all projects
- **Request:** to request a project (part of the stage gate process described)
- **Feedback:** to submit feedback about a particular project (intended for customers)
- **Update:** to update the status of a particular project (intended for project managers)

Other features on the landing page include the nav-bar with the 'Home' option and search function. The nav-bar has been developed using twitter bootstrap and is present on all pages once the user is logged in. The search function allows the user to search for a particular project.





Project Request All ideas, requests and suggestions are reviewed by the team.

Please complete the fields below to submit an idea or project request. Once this request is submitted, the project team will review and progress your request through the Stage Gate process. For more information about this process, please follow the link.

[Back](#)

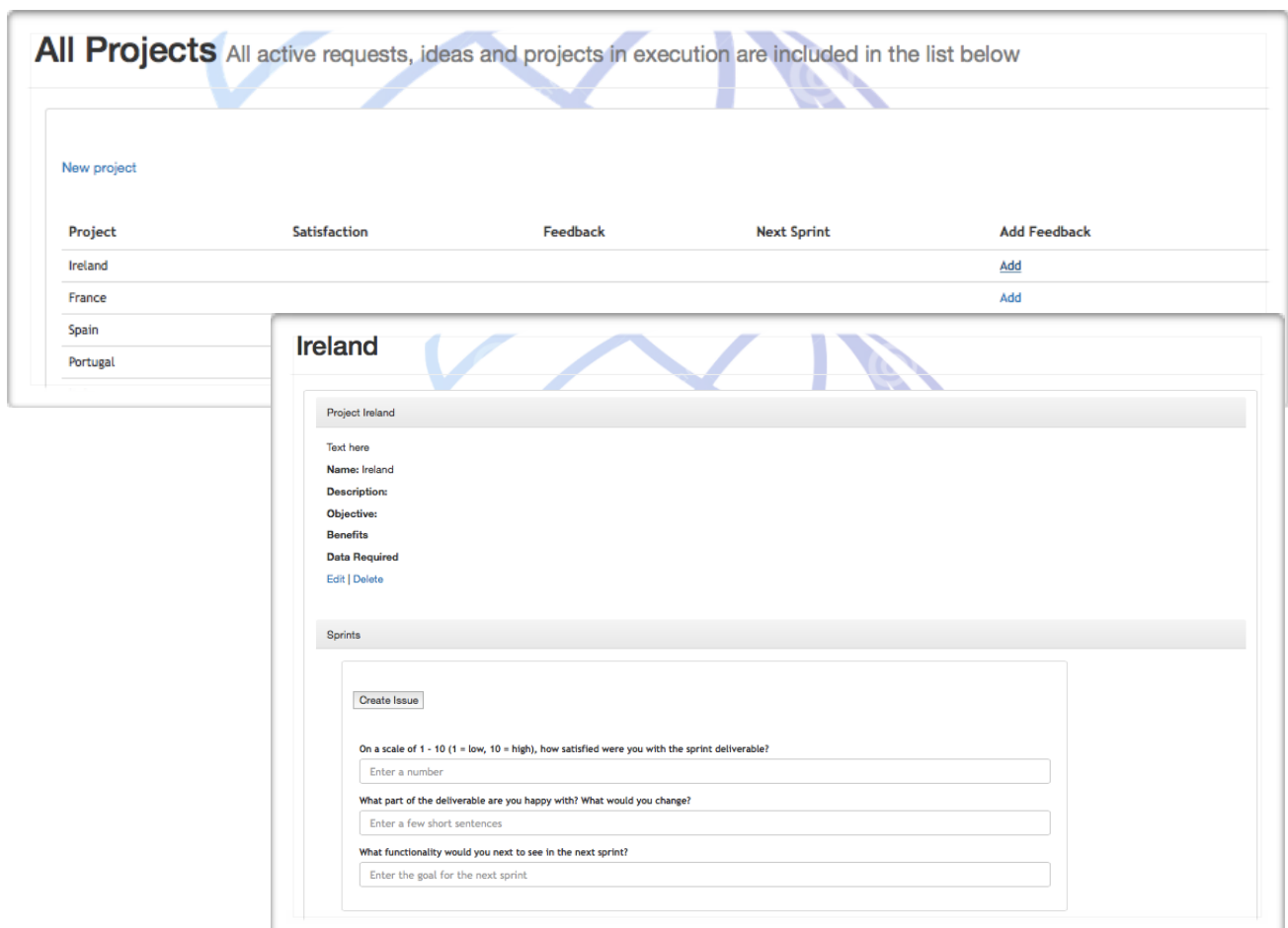
Project Title A short title for the project	Data Required Data Required to complete the project
Description Description	Legal and Compliance Restrictions Holiday Allowance
Objectives Objectives	Executive Sponsor Executive Sponsor
Benefits Benefits associated with the project	Platform Executive Sponsor

[Create Project](#)

Fig 11 shows the form to be completed to submit a project idea or request. The form is designed in a way that the information captured will allow the stage gate team to review and understand the requirements, as well the potential value and return on investment to be delivered through the request. Once the user completes the form, they will submit the request.

Fig 11: Project Request form

Fig 12 shows how the customer/colleague can navigate to the feedback section, to add feedback about a particular sprint release.



All Projects All active requests, ideas and projects in execution are included in the list below

[New project](#)

Project	Satisfaction	Feedback	Next Sprint	Add Feedback
Ireland				Add
France				Add
Spain				
Portugal				

Ireland

Project Ireland

Text here

Name: Ireland

Description:

Objective:

Benefits

Data Required

[Edit](#) | [Delete](#)

Sprints

[Create Issue](#)

On a scale of 1 - 10 (1 = low, 10 = high), how satisfied were you with the sprint deliverable?

Enter a number

What part of the deliverable are you happy with? What would you change?

Enter a few short sentences

What functionality would you next to see in the next sprint?

Enter the goal for the next sprint

Fig 13 shows how the status of a particular project can be updated. Updates are added by project managers at the end of each sprint

Editing Project Click update project to save your changes

Status update

Project Status

[Back](#) | [All Projects](#)

Fig 13: Adding a status update

Fig 14 shows the summary of projects, grouped by the project attributes such as platform, scrum master and executive sponsor. Once the required metrics are agreed, they will be incorporated into these views. The current charts are visible as placeholders until then.

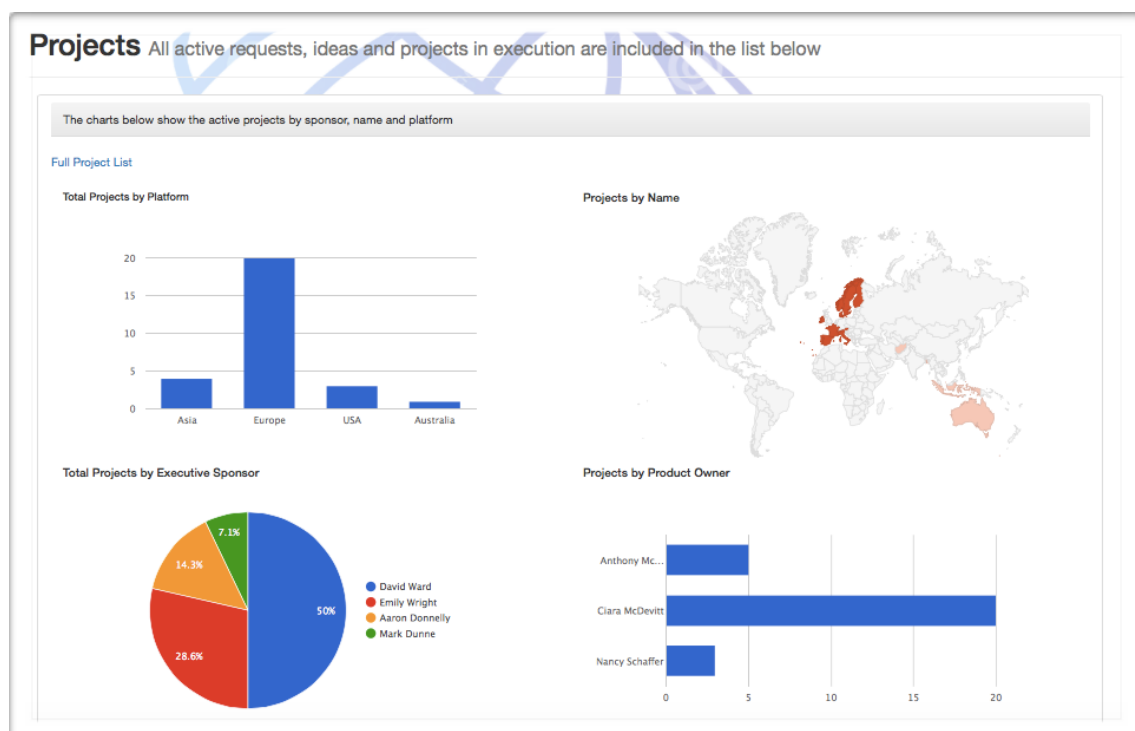


Fig 14: Viewing the project summary

2.6 Customer testing

To understand the success against each of the user requirements, and the likely hood of the intended audience to use the applications, a few employees of the company were asked to carry out testing, and assessments. The below table shows a blank template that is to be completed by each of the users asked. The users included PMO manager, marketing coordinator, project manager, chief operating officer and a customer (colleague external to the department). Results will be incorporated into enhancements of the project. These surveys will be redistributed in 12months, to ensure the anticipated benefits are realised, and any application enhancements are noted in the backlog of improvements and incorporated into future enhancements.

“Truly impressed by this application, once the planned enhancements are implemented, it will be rolled out to other departments within the organisation.”

Ben Blake, Chief Operating Officer

Question	1 (Low) - 10 (High)	Comments
How well does this application meet your requirements and alleviate any pain points you are currently experiencing?		
How would you score the general look and feel of the application? Is it brand compliant?		
Would you recommend this application to a colleague?		

“The look and feel of the application is very professional, allowing our department stand out for the best reasons”

William Henry, Marketing Manager

3. Conclusions

Magile will bring many advantages to TheSun @ TheGlobe Inc. With the aim of getting the right information to the right people at the right time, the application will allow senior managers to login and view what's going on across the department at any point in time. They can currently see basic metrics associated with the projects to include the number of projects associated with platforms or executive sponsors. With further product enhancements, senior managers will be able to gain deeper insights to the projects and associated developments within the departments, for example, the quality of deliverables, the issues or backlog items associated with sprints.

Additionally the scaffold to allow users to be added to the system has already been developed (although this was not part of the initial requirements), and is not ready to be deployed, the project management office will be able to add resource profiles, so the capacity within the centre is clear, with metrics associated with the resources. This will bring a myriad of benefits, to include end of year reviews - reflecting back on what employees have achieved over the year, resource capacity planning - to understand tasks currently assigned to employees, and also as a skills profile. Often when new projects, particularly those with a substantial degree of innovation, are initiated, it's important to understand who within the department has specific skills e.g. can speak Spanish, can develop in Python etc, even though they may not be working in a team associated with these skills. That way, TheSun can ensure optimal use of resources.

As a disadvantage, the system will be another source of the truth, and another system to maintain access, product enhancements etc. Where new joiners will have to be added, leavers will have to be removed and system access will have to be reviewed regularly. However, the benefits delivered by the system and the associated time gain, especially from resources within the project management office most definitely outweigh these. Additionally, the system will need to be reviewed by legal and compliance to ensure it complies with their standards, and the level of access to each of the different parts of the application are as they should be. The active admin gem could be used for the development of this functionality.

Additionally users may need to be trained, and given that senior management aren't necessarily viewing metrics that they have been receiving to date from the project management, the benefits of such metrics, and the aim of showing these vs. the more traditional metrics associated with waterfall project management will need to be communicated to senior management. One threat is that senior management will not buy into these enhancements and continue requesting updated view of historical materials produced by the project management office.

As the organisation continues to roll out agile project management across departments, this tool could potentially be adopted, clearly generating benefits for executives, needing views of the entire organisation to make the right decisions, and ensure they remain aligned to strategic initiatives, adding maximum value for clients.

The above points, as well as additional strengths, weaknesses, opportunities and threats are summarised in the SWOT analysis below.

Strengths	Weaknessess
<ul style="list-style-type: none">• Management can view the entire portfolio of projects• There is a forum to add project requests - promoting innovation in the copy, where requestors can view the latest status of the projects they have requested, and therefore be more involved in the product development process, where their requirements are captured earlier in the project, which increases the chances of project success, benefits realisation and maximum return on investment• There is huge room for improvement due to the scalability of the application• The system mitigates the need to expand the number of JIRA licenses, as senior management can view project related information through the application rather than login to JIRA• The application is brand compliant, something that could not be fully achieved through a vendor system• There is no dependency on vendors, the system can be fully configured and designed according to the departments requirements, as required	<ul style="list-style-type: none">• Not yet integrated with JIRA - and therefore does not provide the needed realtime view• Some bugs remain in the system• Access to the system is open and should be reviewed• Requires resource training and change of management mindset to be 100% successful
Opportunities	Threats
<ul style="list-style-type: none">• Build solutions to solve additional business problems through enhancements, for example:<ul style="list-style-type: none">• Allow employees to request holidays (current a manual process, with an excel maintained by the admin team)• Build metrics associated with employee satisfaction and relate them back to project teams - to build early warning indicators for employee loses, or to understand where employee morale boosting exercises should be concentrated• Send automated emails back to project requestors once the status of their project or request has been updated• Build metrics associated with	<ul style="list-style-type: none">• The security associated with the system - a legal, security and compliance audit should be done to understand the system weaknesses and mitigate risks associated with reputation loss or competitor access to employee, project and portfolio related decisions• All information is in the one place, and therefore the access to the system must be careful managed, not all users should be able to access the databases, and do bulk downloads of system data

4. Further development or research

The system has been developed in such a way that it can be scaled to meet increased requirements from the business, or to cope with demands from an increased number of users, and therefore the system has high potential for development. Further areas for application development include:

- Capturing employee satisfaction - and associating metrics with project teams, to get fact based insights and understand areas of weaknesses
- Finance and Invoicing - develop the application to include an area for finances, invoices and therefore management can get a realtime view of the spend associated with the projects, platforms, people and teams.
- Holiday tracking - this is a time intensive ask currently being done by the admin team through an excel spreadsheet, which will naturally involve human error, and a high cost - associated with miscalculated annual leaves days.
- Partner and vendor health. Given the reliance on some vendors being used by the department, it's important that they know the health of their partners and vendors, aiding them to make decisions such as increased investment in their vendors. Public sources of information could be used for this, sources such as Bloomberg would provide data that could generate metrics to spot partners that are getting into 'danger', and therefore the organisation can react accordingly.
- HR system - Integration with the HR system would mean that senior management could view employee profiles, skills, current projects and holidays all in one place. This would help in making decisions regarding the allocation of employees to projects and the capacity of employees.
- Newsfeed - A newsfeed from an external source could help with early identification of competitor threats. Where the tool would scan the news to find similar products being released by competitors that project teams should be aware of.
- Export function - The system should provide PowerPoint export summaries of projects and platforms to be used in management meetings.

In conclusion, the project has been successful, it has achieved the functionalities required, but the main success has been the scalability associated. Through Heroku the system infrastructure can expand as required and to meet the evolving needs of both the department and the organisation. Given the number of potential enhancements, the discussed should be reviewed and prioritised according to the value to be generated from them. This will ensure the optimal evolution of the system, and as with all business change, it is important to keep the relevant stakeholders involved in these decisions and the through the continuing development of the application.

5. References

Websites:

CAPRANI, D.

D-Log Technical Report

In-text: (Caprani, 2013)

Bibliography: Caprani, D. (2013). D-Log Technical Report. [online] www.deesee.ie. Available at: http://deesee.ie/Publications/D-Log_Tech_Report.pdf [Accessed 1 Mar. 2015].

GUIDES.RUBYONRAILS.ORG

Ruby on Rails Guides

In-text: (Guides.rubyonrails.org, 2015)

Bibliography: Guides.rubyonrails.org, (2015). Ruby on Rails Guides. [online] Available at: <http://guides.rubyonrails.org> [Accessed 7 Feb. 2015].

HEROKU.COM

Heroku | Cloud Application Platform

In-text: (Heroku.com, 2015)

Bibliography: Heroku.com, (2015). Heroku | Cloud Application Platform. [online] Available at: <https://www.heroku.com> [Accessed 8 Apr. 2015].

-

6. Appendix

The additional pages include other documentation associated with the project development, delivery, deployment and design. These include:

- Project Proposal
 - Requirements Specification
 - Monthly Journals
-