Project: ACTIONit

# Requirements and Analysis Document

[By: Team IDEAS PT02]

Louis Ng

Project Manager

Nicol Loh Chuen

Communications Officer

James Lee

Software Coordinator

Zhao Peng

Software Coordinator Assistant

Norman Seck

Librarian / Secretary

Abel Law

User Interface Coordinator

Table of Contents

[Requirements and Analysis Document 1](#_Toc474684567)

[1.0 Version history 3](#_Toc474684568)

[2.0 Executive Summary 4](#_Toc474684569)

[3.0 Client’s Organisation Background 4](#_Toc474684570)

[4.0 Introduction 5](#_Toc474684571)

[4.1 Objective and goals of the proposed system 5](#_Toc474684572)

[4.2 Problems face by the organisation 6](#_Toc474684573)

[4.3 Discussion of methodology used 6](#_Toc474684574)

[4.4 Benefits of the proposed system 8](#_Toc474684575)

[5.0 Outline of proposed solutions to solve the problems 8](#_Toc474684576)

[6.0 Functional Requirements 10](#_Toc474684577)

[7.0 Non-functional requirements 11](#_Toc474684578)

[8.0 Diagrammatic representation of requirements 13](#_Toc474684579)

[8.1 Activity Diagram 13](#_Toc474684580)

[8.2 Use Case Diagram 14](#_Toc474684581)

[8.3 List of use cases 15](#_Toc474684582)

[8.4 Fully documented use case description 16](#_Toc474684583)

[8.4.1 Use-case description for Register account 16](#_Toc474684584)

[8.4.2 Use-case description for Login Account 17](#_Toc474684585)

[8.4.3 Use-case description for Forgot Password 18](#_Toc474684586)

[8.4.4 Use-case description for Logout 19](#_Toc474684587)

[8.4.5 Use-case description for Activate Account 20](#_Toc474684588)

[8.4.6 Use-case description for Edit Profile 21](#_Toc474684589)

[8.4.7 Use-case description for View Profile 21](#_Toc474684590)

[8.4.8 Use-case description for Add user 22](#_Toc474684591)

[8.4.9 Use-case description for Delete User 23](#_Toc474684592)

[8.4.10 Use-case description for Deactivate User 24](#_Toc474684593)

[8.4.11 Use-case description for Activate User 25](#_Toc474684594)

[8.4.12 Use-case description for Create Project 26](#_Toc474684595)

[8.4.13 Use-case description for Update Project 27](#_Toc474684596)

[8.4.14 Use-case description for Delete Project 28](#_Toc474684597)

[8.4.15 Use-case description for View Project 29](#_Toc474684598)

[8.4.16 Use-case description for Add Team Member 30](#_Toc474684599)

[8.4.17 Use-case description for View Team Member 31](#_Toc474684600)

[8.4.18 Use-case description for Remove Team Member 32](#_Toc474684601)

[8.4.19 Use-case description for Create Activity 33](#_Toc474684602)

[8.4.20 Use-case description for Update Activity 34](#_Toc474684603)

[8.4.21 Use-case description for Delete Activity 35](#_Toc474684604)

[8.4.22 Use-case description for Create Task 36](#_Toc474684605)

[8.4.23 Use-case description for Update Task 37](#_Toc474684606)

[8.4.24 Use-case description for Delete Task 38](#_Toc474684607)

[8.4.25 Use-case description for View Task 38](#_Toc474684608)

[8.4.26 Use-case description for Upload Attachment 39](#_Toc474684609)

[8.4.27 Use-case description for Remove Attachment 40](#_Toc474684610)

[8.4.28 Use-case description for Download Attachment 41](#_Toc474684611)

[8.4.29 Use-case description for Email User 41](#_Toc474684612)

[8.4.30 Use-case description for Project Search 42](#_Toc474684613)

[8.5 Models for processes in the system 43](#_Toc474684614)

[8.5.1 Context Diagram 43](#_Toc474684615)

[8.6 Models for data in the system 44](#_Toc474684616)

[8.6.1 Level 0 Entity-Relationship Diagram 44](#_Toc474684617)

[8.7 CRUD Matrix 45](#_Toc474684618)

[9.0 Working with External Party 46](#_Toc474684619)

[9.1 Chat Group Communication Channel 46](#_Toc474684620)

[9.2 Important Email Correspondences Between Teams: 47](#_Toc474684621)

[9.3 Challenge Faced 49](#_Toc474684622)

[10.0 Conclusion 49](#_Toc474684623)

# **1.0 Version history**

The following table reviews all version changes made to this document.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Version Number | Description of change | Author(s) | Approved by | Date |
| Version 1.0 | * Initial version of documentation | Louis Ng  Nicol Loh  Norman Seck  James Lee  Zhao Peng Abel Law | Louis Ng | 25th Jan 2017 |
| Version 2.0 | * Added initial version of activity diagram * Added list of use cases | Nicol Loh | Louis Ng | 28th Jan 2017 |
| Version 3.0 | * Added use case descriptions * Edits to content of R&A | Nicol Loh Norman Seck | Louis Ng | 30th Jan 2017 |
| Version 4.0 | * Finalised use case list and descriptions * Finalised activity diagram | James Lee Zhao Peng Abel Law | Louis Ng | 2 Feb 2017 |
| Version 3.0 | * Added in CRUD matrix * Added in ERD | Nicol Loh Norman Seck | Louis Ng | 5th Feb 2017 |
| Version 4.0 | * Amended Use Case list to reflect combined use cases between teams * Added concept art of user dashboard interface | Nicol Loh  Norman Seck  Abel Law | Louis Ng | 8th Feb |
| Version 5.0 | * Amended ERD * Added in context DFD | Norman Seck  Zhao Peng  James Lee | Louis Ng | 10th Feb 2017 |
| Version 6.0 | * Amended Use Case list to reflect only Team IDEAS use cases | Nicol Loh  Norman Seck | Louis Ng | 11th Feb 2017 |
| Version 7.0 | * Finalised the document | Nicol Loh Norman Seck | Louis Ng | 12th Feb 2017 |

# **2.0 Executive Summary**

This Requirement and Analysis document states the purpose of the system, lists down its requirements and scope, as well as showing how the system will work. It seeks to give the reader the context of the problem domain, solutions to the problems, as well as diagrammatic representations on how users will be interacting with the system, how the activity flow from users will be like, the authorisation levels of different kind of users, as well as how the various entities will interact with one another.

Sections three and four provides the context of the clients organisation background and the problems it faces, an introduction to the proposed system Huddle, describes the objectives and goals of the proposed system, the type of methodology used and its rationale, and how the proposed system will benefit the client.

Section five outlines the solutions to the problems stated, giving a conceptual view of how the proposed system will work, as well as some initial concept of how the proposed system will look like.

Sections six and seven list down the functional and non-functional requirements of the proposed system, quantifying the features the system will need to have, and defining how the system will behave.

Section eight shows the diagrammatic representations of how the system will work, and includes use case diagrams and descriptions, an activity diagram, a context diagram, an entity relationship diagram, as well as a CRUD matrix. All to allow for the concept and idea of the proposed system to be communicated effectively.

Section nine lists down the process of working with an external party to delivering one deliverable, the problems faced, as well as the solution(s) that were proposed and the results of such action(s).

This document will end in section ten with a conclusion, giving a short overview on why the proposed system Huddle will assist in the solving of the client’s existing problems.

# **3.0 Client’s Organisation Background**

The client is Professor Peter Cole, a senior lecturer in the School of Engineering and Information Technology who is representing Murdoch University. Murdoch University’s inception began in 1973 and is currently based in Perth, Australia (Murdoch University, 2016). It prides itself as a research led institution, and is currently ranked 401 to 500th in the world (Times Higher Education, 2017).

Murdoch has over 2,000 staff and 30,000 students from around 100 countries studying both in Perth and offshore, and offers more than 80 majors across multiple industries with each of the degree courses at Murdoch will require students to complete a major project. With the huge amount of projects that the Murdoch University staff have to manage, a solution called ActionIt is used as a Project Management Tool to allow for project managers to create projects and its tasks, as well as assign team members to the said tasks, with deliverables in the task to be submitted as files (spreadsheets, slides, PDF etc). The current version called UpBox allows for such functionality, however client wishes to enhance the project management tool further and has tasked Team IDEAS and Team Evolve to come up with enhanced features to expand on the current solution as a single deliverable.

# **4.0 Introduction**

A Project Management Tool (PMT) will allow organisations to better manage the projects it has to complete in the course of their operations. Using such tools will allow for better tracking, allowing Project Managers to be better able to view what the team’s progress is, make better decisions on allocating resources, know which team member is lagging behind in deliverables and to be able to focus on him/her, and in general assist the team to stay the course to complete their objectives and KPIs.

This can bring about tangible benefits such as effective cost control, and increased revenue; as well as intangible benefits such as increased staff morale for the successful implementation of a project.

Our solution, Huddle, aims to rethink how the system does the entire process of managing projects. This rethink looks at how the solution will be applied to users, and to make it a simpler, faster, and easier way for users to get started. It looks at how organisations have a hierarchy, and how project management can be done higher up the hierarchy level, and to manage the receipt of all deliverables, all within one solution.

## 4.1 Objective and goals of the proposed system

The purpose of this system is to enhance the capabilities of ActionIt (UpBox), to contain the current functionalities of the system, and enhance it further. General goal is to bring greater stability to the system, improve on the usability, easing up on the workload required by system administrators, and adding on improvements and features to assist team leaders to manage their projects. The solution will assist team leaders in their project management by:

* Still have the existing functionality of ActionIt (Upbox)
* Changing the way, the solution will interact and allow users to create their projects by allowing automated registration.
* Allowing all users to have the potential to be both project owners and team members
* Removal of the waiting time for users waiting to get authorisation from system admin for user creation.
* Improved visibility metrics via a dashboard to allow users to get most relevant information easily at a glance.
* Allow for better self-management of users to create their teams and pull in team members
* Implementation of security features such as captcha boxes and password complexity requirements
* Addition of features such as task lists to aid in productivity, as well as comment box and email notifications for easy communication
* Improved user interface and aesthetics

## 4.2 Problems face by the organisation

During the initial discussion with the client, the client was stating that while the current version of ActionIT (UpBox) was good, he is hoping to see expansion on the features of the solution. Key to this is to expand on the current solution with features that will aid in project management. As it is, the current solution allows for the creation of project owners and users, and allow for project owners to pull in team members into their teams, assign tasks to them, and allow the team members to upload their deliverables.

Some limitations to this current system include:

* Heavy work load on system administrator to manually create and delete users
* Lack of visible metrics to allow project owners and team members to have an over view of the projects they are involved in
* Lack of scalability as current system relies heavily on system administrator’s efforts
* Increased waiting time for project owners for the creation of users
* Lack of a way for non-technical users to update the content on the webpage
* Lack of security features which make the solution more susceptible to breaches
* Lack of bulk exporting features which take up time and may cause human error form missing out on specific documents

## 4.3 Discussion of methodology used

Through the team discussion, it is agreed upon that Team IDEAS will work towards initial scope definition, locking it to prevent scope creep, and then develop Huddle in iterations, with the end of each iteration allowing for testing to be done.

In this, the closest methodology to be used will be the Rational Unified Process (RUP), and we will be borrowing concepts from the RUP for this project management. However it must be noted that due to time constraints, strict adherence to the RUP is not possible, and we have taken components which are applicable to our project development; although it can be said that this project development can be considered as an iteration within the RUP methodology, with future development continuing on with other iterations.

The RUP breaks software development into four phases of inception, elaboration, construction, and transition (Rational Software, 1998). In the inception phase, the scope of the project is determined, and the definition of all actors who interact with the system, all use cases, success criteria, and milestones. A good example for this is the creation of the Requirements and Analysis (R&A) document as well as the Project Management Plan (PMP).

In the elaboration phase, the problem domain is analysed, and the software architecture is established. Potential problems are identified and eliminated or managed. There is clarity on the scope of the project, as well as the vision of what the end product will be like. In our case, the development team has analysed the CETO server they have to work with and choose a framework that will be usable to each of them. Moreover, how the end product will be like is also communicated frequently to ensure that all are on the same page. Collaborative tools such as folders in the cloud allow for easy access across all members in the team. In this phase, use case descriptions are developed, non-functional requirements identified, and an architectural prototype is generated. For this project, the development of a landing page will be one of the items on top of the development agenda, to allow for a larger base of testers to begin testing.

In the construction phase, all other components and features will be developed and integrated into the software. This phase will also allow for parallel development, which will be used for development of the features to be separated out into two separate teams. In this phase, each feature developed and integrated into the software will be tested as well, before final release to the client and public.

In the transition phase, future development for the project can be done after it is determined that the system is stable, and the bugs are corrected. This will be for future features to be added into the system to allow for improvements to be made.

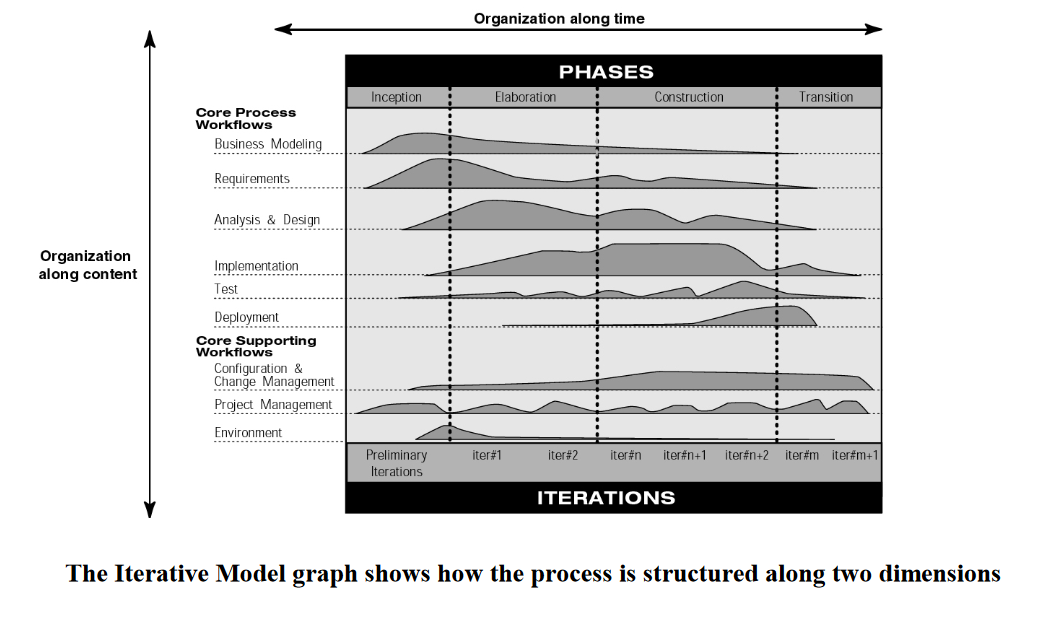


Figure 1 Graph showing the amount of work done in each phase (Rational Software, 1998)

## 

## 4.4 Benefits of the proposed system

Huddle seeks to allow for greater productivity among users, while being a simple and elegant concept for multiple levels of project management and the submission of deliverables. This is done by:

* Automating the registration process for users
* Allowing users to have the potential to be both project owners or team members concurrently
* Allows project owners to create multiple projects that can run concurrently
* Allows both project owners and team members to have access to a personal dashboard, allowing them to have easy access to information such as a task list with due dates (soonest due date on the top) to allow for easy tracking
* Addition of project and task visibility metrics such as project completion %
* Enhanced communication features which allow for better communication for specific tasks

# **5.0 Outline of proposed solutions to solve the problems**

For the problems highlighted above, it can be seen that the current solution ActionIT (UpBox), while are functioning; can be greatly enhanced to assist users in managing a team, or participating in one. Aside from minor bugs and inconsistencies, we felt that how the solution managed its users was a key hindrance in scalability in terms of efforts required. Reworking on the role of a user, we have concluded that in order for the solution to run effectively and efficiently, the requirement of control from system administrators will need to be reduced to strictly case by case scenarios, with the majority of the time allowing the system to run autonomously. This allows improved scalability as manual labour is not required in the creation of users. Moreover, a two-tiered approach to project management, combined with allowing users to be both project owners and team members at the same time gives the solution a flexibility for multiple tiers of project management.

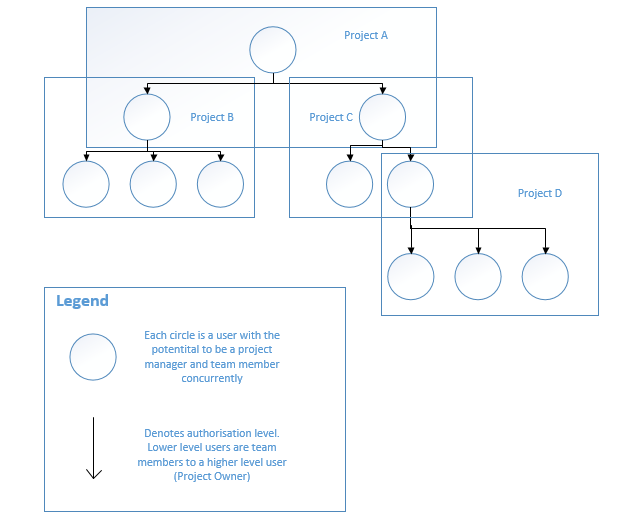


Figure 2 Illustration showing how Huddle can be used in a hierarchy (Source : Team IDEAS)

Figure 2 above shows an illustration of this concept. An example will be in an organisation context, where the top layer of Project A is the General Manager of a company, with the two vice presidents reporting to him as team members. The vice managers in turn create their own projects to include the managers under them (Projects B and C). And the managers include their staff (Project D). at each layer, the user can be both a project owner and team member, simultaneously reporting upwards, and monitoring work progress downwards.

Further investigation into the solution allowed us to understand that while the primary aims of allowing project leaders to distribute work and allow project members to upload / submit deliverables were met; much more could have been done in the visibility and displaying of information which users will require access to. A dashboard to show different metrics and visualisation tools can be added to allow for users to keep track of the tasks and deliverables deadlines, as well as show at a glance metrics such as project completion percentage, and project information.

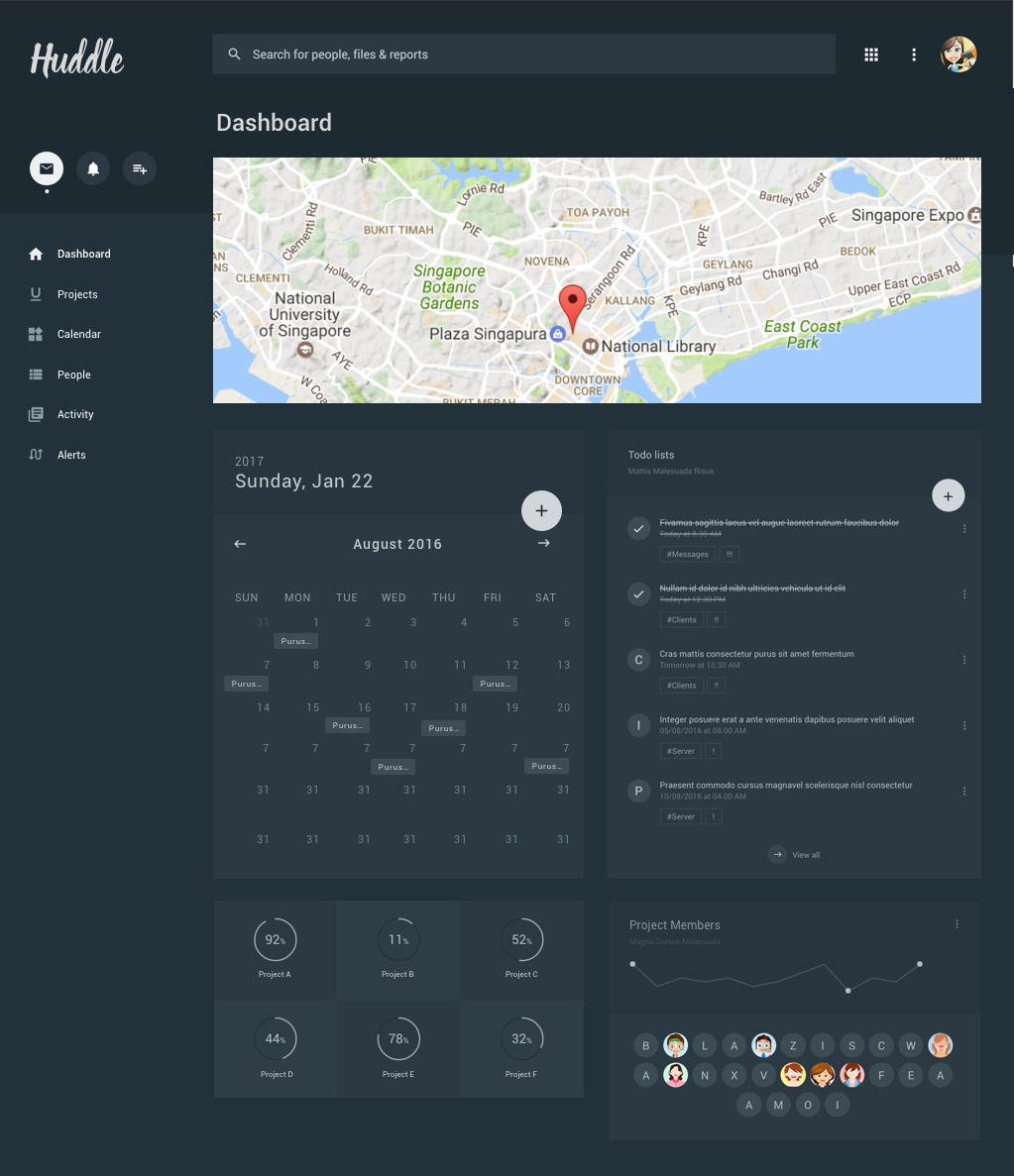


Figure 3 Initial concept of dashboard view. Source : Team IDEAS

Furthermore, while we understand that there are a lot of communication tools as options for users, having one within the solution will greatly help with productivity by reducing the need for users to switch between different tools, allowing for issues pertaining to specific items or tasks to be communicated clearly and effectively. It is not built to be a messaging tool, to allow chats, but rather to allow the communication between project leaders and members to highlight certain things, which allow the affected user to have clear information pertaining to a specific task at one place.

Additionally, team members can also view each other’s task and deliverables in order to facilitate transparency within the team for instances where one task is dependent on the completion of another. Team members can check each other’s tasks for better resource and time management, while authorisation to delete and edit will only be given to the project leader, and the team member(s) whom the task belongs to.

Lastly the user interface and user experience could have been enhanced greatly to allow for the solution to be easier to use. A feature to download the respective documents submitted as deliverables from different levels was added. Project Owners can choose to download all deliverables up to date for a particular project, download all deliverables by task, or by team member. A report generating feature also can be added to allow for the exporting of reports in excel format, to allow for project owner’s use for their own reporting requirements external to the solution.

# **6.0 Functional Requirements**

Functional requirements are described as the actions a system being built must do. It can be in form of features that the system needs to have, like the auto computation of salary in a payroll software (Goldsmith, 2009). Typical functional requirements include business rules, authorization levels, legal or regulatory and certification requirements among others (Eriksson, 2012). As at this stage the project is largely still in the scope definition stage, the below list will show what Huddle is expected to do, and what features it should have:

* Huddle shall have a content management system (CMS)
* Huddle shall have different authorization levels for roles of System administrator, user, project owner, team member
* Huddle shall have a system administrator role that has access to the CMS
* Huddle shall allow system administrators to create and delete users via the CMS
* Huddle shall allow system administrators to manage website content via the CMS
* Huddle shall allow users to register for its service using automated email verification
* Huddle shall allow users to view and edit their own user profiles
* Huddle shall allow users to have access to a dashboard where a task list, project completion metrics, projects list, history can be accessed
* Huddle shall allow users to create, read, update, and delete projects and be project owners
* Huddle shall allow project owners to input details of a project
* Huddle shall allow project owners to search for and include team members into their projects
* Huddle shall allow for project owners to create, read, update and delete tasks
* Huddle shall allow for a project owner to set and edit the status of a project
* Huddle shall allow for project owners to input task details
* Huddle shall allow for project owners to assign task(s) to team member(s)
* Huddle shall allow for project owners to communicate with team members via comment box in task
* Huddle shall allow for project owners to download all documents submitted as deliverables in a project
* Huddle shall allow for project owners to set and edit the status of a task
* Huddle shall allow for users to accept an invitation from a project owner and be a team member
* Huddle shall allow for team members to view tasks he/she is allocated in the particular project
* Huddle shall allow for team member to view other team member’s task(s)
* Huddle shall allow for team members and project owners to communicate with each other via comment box in task
* Huddle shall allow for team members to upload document files as deliverables
* Huddle shall allow team members to edit and delete submissions before task is closed and submission closes

# **7.0 Non-functional requirements**

Non-functional requirements are the characteristics or behaviour of a system being built. It defines how a system should behave, and how the system can be measured (Eriksson, 2012). An example of a non-functional requirements for a payroll system is that the payroll software must be accurate. As such, the non-functional requirements of Huddle are:

* Huddle shall be compatible with major browsers, in particular Firefox, Chrome, and Internet Explorer 10.0 and above
* Huddle shall employ security techniques for users
* Huddle shall be scalable
* Huddle shall be integrated into the Murdoch university CETO server
* Huddle shall have an acceptable loading time
* Huddle shall have improved user interface and user experience

# **8.0 Diagrammatic representation of requirements**

## 8.1 Activity Diagram

Figure 4 Huddle Activity Diagram. Source : Team IDEAS

Figure 4 shows the activity flow for Huddle by the actors System Administrator, System, and Users.

## 8.2 Use Case Diagram

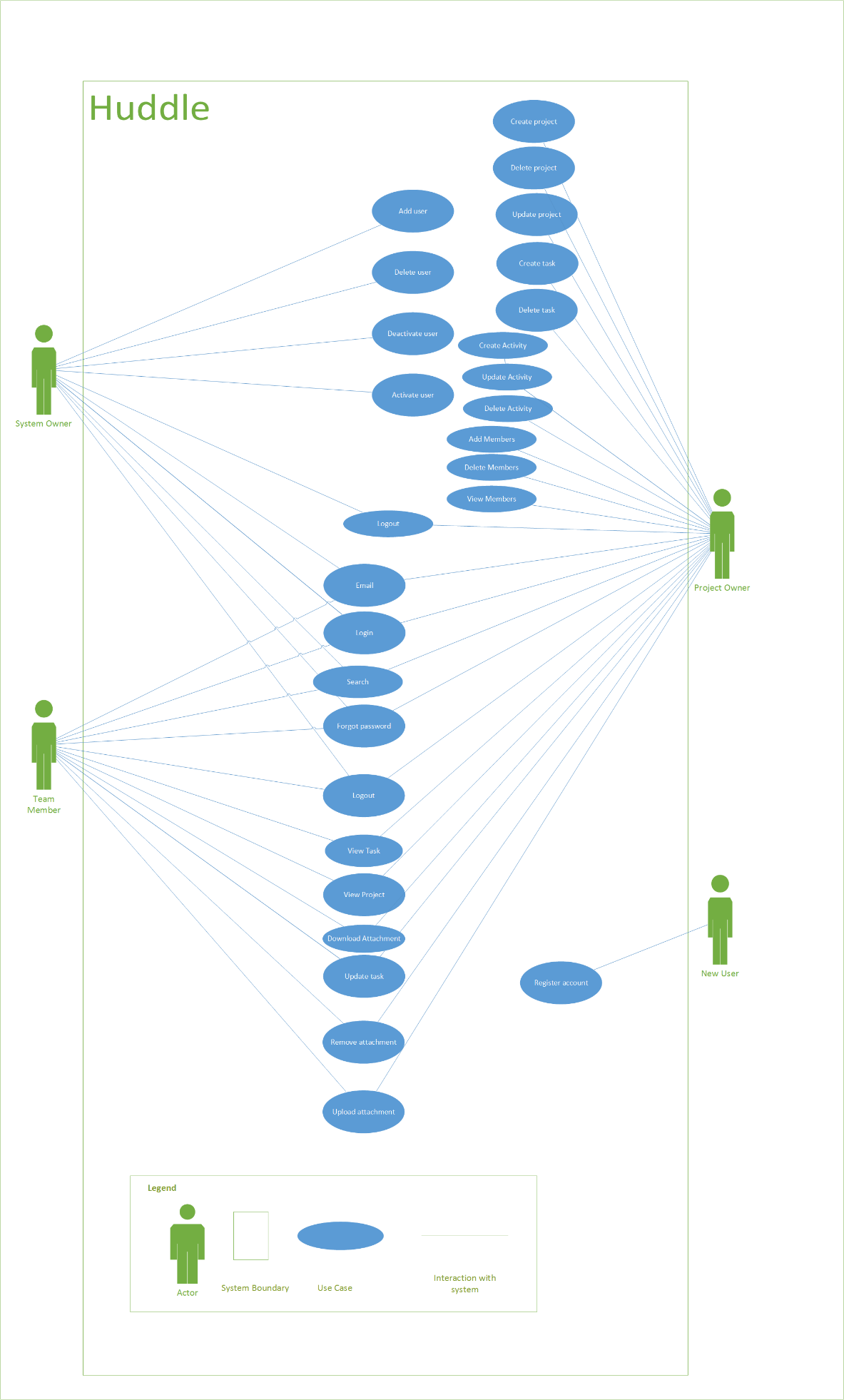


Figure 5 Use case diagram. Source : Team IDEAS

*Figure 5 shows the use case diagram for Huddle. It identifies 4 main actors, the System Owner, New User, Project Owner, and Team Member; and how they interact with the system.*

## 8.3 List of use cases

|  |  |  |  |
| --- | --- | --- | --- |
| S/N | Use case | Use case description | Action person |
| UC-01 | Register account | Allows new user to register account | New User |
| UC-02 | Login account | Existing user to login to account | Project Owner, Project Member, System Owner |
| UC-03 | Forgot password | Existing user to reset password for login | Project Owner, Project Member, System Owner |
| UC-04 | Logout | Logout | Project Owner, Project Member, System Owner |
| UC-05 | Activate Account | Owner to activate accounts | Owner |
| UC-06 | Edit profile | Allows existing users to edit profile details | Project Owner, Project Member, System Owner |
| UC-07 | View profile | Allows existing users to view their profile | Project Owner, Project Member, System Owner |
| UC-08 | Add user | System Owner to add user | System Owner |
| UC-09 | Delete User | System Owner to delete user | System Owner |
| UC-10 | Deactivate User | System Owner to deactivate user | System Owner |
| UC-11 | Activate User | System Owner to activate user | System Owner |
| UC-12 | Create project | Existing user to create project | Project Owner, Project Member |
| UC-13 | Update project | Existing user to update assigned or owned projects | Project Owner, Project Member |
| UC-14 | Delete Project | Project Owner to delete project(s) | Project Owner |
| UC-15 | View Project | To allow members to view projects | Project Owner, Project Member |
| UC-16 | Add team member | Project Owner to add members into the project | Project Owner |
| UC-17 | View Team Members | Existing users to view team members in the project(s) | Project Owner, Project Member |
| UC-18 | Remove Team Members | Project owner to remove members from project | Project Owner |
| UC-19 | Create Activity | Project Owner to create activity | Project Owner |
| UC-20 | Update Activity | Project Owner to update activity name | Project Owner |
| UC-21 | Delete Activity | Project Owner to remove activity | Project Owner |
| UC-22 | Create task | Project owner to create task | Project Owner |
| UC-23 | Update task | Project owner and project member to update the task | Project Owner, Project Member |
| UC-24 | Delete task | Project owner to remove task | Project Owner |
| UC-25 | View Task | Project owner and project member to view task | System Owner, Project Owner, Project Member |
| UC-26 | Upload attachment | Project owner and project member to upload attachment | Project Owner, Project Member |
| UC-27 | Remove attachment | Project owner and project member to remove attachment | Project Owner, Project Member |
| UC-28 | Download attachment | Project owner and project member to download attachment | Project Owner, Project Member |
| UC-29 | Email user | Existing user to send emails to another user | System Owner, Project Owner, Project Member |
| UC-30 | Project Search | Allows existing user to search for details of project | System Owner, Project Owner, Project Member |
| UC-31 | Assign project | Allow System Owner to assign projects with project owner | System Owner |

## 8.4 Fully documented use case description

### 8.4.1 Use-case description for Register account

|  |  |  |
| --- | --- | --- |
| **Use-Case ID:** | UC-01 | |
| **Use-Case Name:** | New User Registration | |
| **Participating actor(s):** | User | |
| **Description:** | A user registers for a new account via register button/link in login | |
| **Precondition:** | Huddle must be accessible | |
| User must have access to website | |
| User must input the correct data for captcha security check | |
| **Typical course of events:** | **Actor Action** | **System Response** |
| 1. User accesses login / register page |  |
| 2. User clicks on register button/link | 3. Huddle shows register page with registration form |
| 4a. User clicks cancel | 5a. Huddle shows login / register page |
| 4b. User inputs required data into registration form and captcha box and clicks "Register now" button | 5b. Huddle checks user has input correct data for captcha. If it is incorrect, displays a "Security check failed. Please try again." message and returns to user input registration form with all data intact (4b). |
|  | 6b. Huddle checks if username input is currently in system. If there is existing identical username, will display "Username is taken. Please enter another." and returns to user input registration form with all data intact (4b). |
|  | 7b. Huddle saves registration form information into database. If successful, displays "User account created." and will display user email verification page. If unsuccessful, will display "User creation error. Please try again." message and returns to user input registration form with all data intact (4b) |
|  | 8b. Huddle sends email with account verification instructions to user |
| 9b. User accesses email account and clicks on verification email link | 10b Huddle receives acknowledgement and updates user’s status to "verified" and saves status into database |
|  | 11a. Huddle shows user verification successful page with go to user dashboard option |
| 12a. User clicks on go to user dashboard | 5. Huddle checks if user account is deactivated. If deactivated, show account deactivated page with reactivation instructions. |
|  | 6. Huddle displays user dashboard |
| **Conclusion:** | User account created by user | |
| **Post condition:** | User will have access to User dashboard | |

### 8.4.2 Use-case description for Login Account

|  |  |  |
| --- | --- | --- |
| **Use-Case ID:** | UC-02 | |
| **Use-Case Name:** | Existing User Login | |
| **Participating actor(s):** | User | |
| **Description:** | An existing user who has already registered logins into Huddle | |
| **Precondition:** | Huddle must be accessible | |
| User must have access to website | |
| User must have already registered for an account | |
| Account must not be deactivated | |
| **Typical course of events:** | **Actor Action** | **System Response** |
| 1. User accesses login / register page |  |
| 2. User clicks on login button/link | 3. Huddle checks if login credentials are correct. If wrong displays "Your user name / password is wrong. Please try again" message, and brings user back to login screen |
|  | 4. Huddle checks if user account is verified. If not verified shows verify email page with verification instructions and resend verification email instruction |
|  | 5. Huddle checks if user account is deactivated. If deactivated, show account deactivated page with reactivation instructions. |
|  | 6. Huddle displays user dashboard |
| **Conclusion:** | Existing user logs in. Huddle prompts the user to relevant steps in order to do successful login | |
| **Post condition:** | Existing user will have access to user dashboard. | |

### 8.4.3 Use-case description for Forgot Password

|  |  |  |
| --- | --- | --- |
| **Use-Case ID:** | UC-03 | |
| **Use-Case Name:** | Forgot password | |
| **Participating actor(s):** | Project Owner Project Member System Owner | |
| **Description:** | An existing user who has already registered logins into Huddle whom forgotten their password and wish to reset it | |
| **Precondition:** | Huddle must be accessible | |
| Must be from UC16 |  |
| User must have access to website | |
| User must have already registered for an account | |
| Account must not be deactivated | |
| **Typical course of events:** | **Actor Action** | **System Response** |
| 1. User clicks on forgot password hyperlink | 2. System to display email field for user to fill in |
| 3. User to enter email | 4. System will perform validation.   4a. If email is found, system will trigger password reset instruction to email and display "Password reset request triggered successfully."  4b. If email not found, system will display "Email not found, try again". Use case repeats at event number 2 |
| **Conclusion:** | Email triggered to user mailbox to reset | |
| **Post condition:** | System will display login page again. | |

### 8.4.4 Use-case description for Logout

|  |  |  |
| --- | --- | --- |
| **Use-Case ID:** | UC-04 | |
| **Use-Case Name:** | Existing User Logout | |
| **Participating actor(s):** | User | |
| **Description:** | An existing user who has already registered logs out from Huddle | |
| **Precondition:** | Huddle must be accessible | |
| User must have access to website | |
| User must have already registered for an account | |
| Account must not be deactivated | |
| **Typical course of events:** | **Actor Action** | **System Response** |
| 1. User click log out page | 2. System to display message "Are you sure to log out?" and display options "Yes" and "No" |
| 3. User clicks "Yes" | 4. Systems displays message "User logged out successfully" and logs out user. |
| **Alternative course of event** | 5. User clicks "No" | 6. System returns to page where user last stopped at |
| **Conclusion:** | Existing user logs out. | |
| **Post condition:** | User be directed to login page. | |

### 8.4.5 Use-case description for Activate Account

|  |  |  |
| --- | --- | --- |
| **Use-Case ID:** | UC-05 | |
| **Use-Case Name:** | Existing User Verifies Email | |
| **Participating actor(s):** | User | |
| **Description:** | An existing user who has already registered verifies email used for registration | |
| **Precondition:** | Huddle must be accessible | |
| User must have access to website | |
| User must have already registered for an account | |
| User must have registered with a valid email account | |
| Account must not be deactivated | |
| **Typical course of events:** | **Actor Action** | **System Response** |
| 1. User accesses login / register page |  |
| 2. User clicks on login button/link | 3. Huddle checks if login credentials are correct. If wrong displays "Your user name / password is wrong. Please try again" message, and brings user back to login screen |
|  | 4. Huddle checks if user account is verified. If not verified shows verify email page with verification instructions and resend verification email instruction |
| 5. User clicks on resend verification email option | 6. Huddle prompts for confirmation of user email. |
| 7. User inputs correct email and / or clicks ok button | 8. Huddle sends email verification link with verification instructions to user email address |
| 9. User accesses email account and clicks on verification email link | 10 Huddle receives acknowledgement and updates users status to "verified" and saves status into database |
|  | 11. Huddle shows user verification successful page with go to user dashboard option |
| 12. User clicks on go to user dashboard | 13. Huddle checks if user account is deactivated. If deactivated, show account deactivated page with reactivation instructions. |
|  | 14. Huddle displays user dashboard |
| **Conclusion:** | Existing user verifies email. User account status changes to "verified" | |
| **Post condition:** | Existing user will have access to user dashboard. | |

### 8.4.6 Use-case description for Edit Profile

|  |  |  |
| --- | --- | --- |
| **Use-Case ID:** | UC-06 | |
| **Use-Case Name:** | Edit profile | |
| **Participating actor(s):** | Project Owner Project Member System Owner | |
| **Description:** | An existing user who has already registered logins into Huddle whom wish to edit their profile | |
| **Precondition:** | Huddle must be accessible | |
| User must have access to website | |
| User must have already registered for an account | |
| Account must not be deactivated | |
| **Typical course of events:** | **Actor Action** | **System Response** |
| 1. User clicks Edit Profile | 2. System to display following fields to change: - Email address - First Name - Last Name - Avatar |
| 3. User to make changes to the existing fields and click "Save" | 4. System will display status "Save successful". |
| **Conclusion:** | User has edited the profile | |
| **Post condition:** | System will display dashboard again. | |

### 8.4.7 Use-case description for View Profile

|  |  |  |
| --- | --- | --- |
| **Use-Case ID:** | UC-07 | |
| **Use-Case Name:** | View Profile | |
| **Participating actor(s):** | Project Owner Project Member System Owner | |
| **Description:** | An existing user who has already registered logins into Huddle whom wish to view their profile | |
| **Precondition:** | Huddle must be accessible | |
| User must have access to website | |
| User must have already registered for an account | |
| Account must not be deactivated | |
| **Typical course of events:** | **Actor Action** | **System Response** |
| 1. User clicks View Profile | 2. System to display following fields to change: - Email address - First Name - Last Name - Avatar - Task List, Project name and due date in table form - Past projects involve |
| **Conclusion:** | Profile has been displayed | |
| **Post condition:** | User profile displayed | |

### 8.4.8 Use-case description for Add user

|  |  |  |
| --- | --- | --- |
| **Use-Case ID:** | UC-08 | |
| **Use-Case Name:** | Create user account in CMS | |
| **Participating actor(s):** | System Owner | |
| **Description:** | System Owner creates user account in CMS | |
| **Precondition:** | Huddle must be accessible | |
| System Owner must access CMS and be logged in | |
| Username is currently not in system | |
| **Typical course of events:** | **Actor Action** | **System Response** |
| 1. System Owner clicks on "User Management" | 2. Huddle displays user management screen |
| 3. System Owner clicks on "Add User" | 4. Huddle displays add user screen |
| 5. System Owner inputs user information form and clicks "Create" | 6. Huddle will check if username exists in database |
|  | 7. If username is not in database, Huddle will save the details in information form into database. If successful, will display "User created" message. If unsuccessful will display "User not created" message |
|  | 8. If username is found to be in database, Huddle will display "Username currently exists. Please choose another one." message |
| **Conclusion:** | A user account is successfully created | |
| **Post condition:** | All user accounts are displayed in the CMS for access by only the system Owner. | |

### 8.4.9 Use-case description for Delete User

|  |  |  |
| --- | --- | --- |
| **Use-Case ID:** | UC-09 | |
| **Use-Case Name:** | Delete User Account in CMS | |
| **Participating actor(s):** | System Owner | |
| **Description:** | System Owner deletes user account in CMS | |
| **Precondition:** | Huddle must be accessible | |
| System Owner must access CMS and be logged in | |
| User account has been created and is currently in system | |
| **Typical course of events:** | **Actor Action** | **System Response** |
| 1. System Owner clicks on "User Management" | 2. Huddle displays user management screen showing list of user accounts |
| 3. System Owner clicks on check box next to user account to select it and clicks on delete button | 4. Huddle displays message "Are you sure you wish to delete this user account?" message |
| 5a. If System Owner clicks on yes button | 6a. Huddle deletes record of user account from system and displays "User account deleted." message |
|  | 7a. Huddle sends an email to user account owner to notify them that their account has been deleted. |
| 5b. If System Owner clicks on cancel button | 6b. Huddle returns back to user management screen showing list of user accounts with check box of selected user accounts still ticked |
| **Conclusion:** | Selected user account deleted. | |
| **Post condition:** | All user accounts are displayed in the CMS for access by only the system Owner. Deleted data are purged from the system. | |

### 8.4.10 Use-case description for Deactivate User

|  |  |  |
| --- | --- | --- |
| **Use-Case ID:** | UC-10 | |
| **Use-Case Name:** | Deactivate User Account in CMS | |
| **Participating actor(s):** | System Owner | |
| **Description:** | System Owner deactivates user account in CMS | |
| **Precondition:** | Huddle must be accessible | |
| System Owner must access CMS and be logged in | |
| User account has been created and is currently in system and is not deactivated | |
| **Typical course of events:** | **Actor Action** | **System Response** |
| 1. System Owner clicks on "User Management" | 2. Huddle displays user management screen showing list of user accounts |
| 3. System Owner clicks on check box next to user account to select it and clicks on deactivate button | 4. Huddle displays message "Are you sure you wish to deactivate this user account?" message |
| 5a. If System Adminitsrator clicks on yes button | 6a. Huddle freezes user account in system from any form of activity and displays "User account deactivated." message |
|  | 7a. Huddle sends an email to user account owner to notify them that their account has been deactivated with re-activation instructions. |
| 5b. If System Owner clicks on cancel button | 6b. Huddle returns back to user management screen showing list of user accounts with check box of selected user accounts still ticked |
| **Conclusion:** | Selected user account deactivated. | |
| **Post condition:** | All user accounts are displayed in the CMS for access by only the system Owner. Deactivated user accounts will not be able to login into system | |

### 8.4.11 Use-case description for Activate User

|  |  |  |
| --- | --- | --- |
| **Use-Case ID:** | UC-11 | |
| **Use-Case Name:** | Reactivate User Account in CMS | |
| **Participating actor(s):** | System Owner | |
| **Description:** | System Owner reactivates a deactivated user account in CMS | |
| **Precondition:** | Huddle must be accessible | |
| System Owner must access CMS and be logged in | |
| User account has been created and is currently in system and is deactivated | |
| User accounts that have been deactivated will have a deactivated status | |
| **Typical course of events:** | **Actor Action** | **System Response** |
| 1. System Owner clicks on "User Management" | 2. Huddle displays user management screen showing list of user accounts |
| 3. System Owner clicks on check box next to user accounts which have a deactivated "status" to select it and clicks on reactivate button | 4. Huddle un-freezes user account in system and displays "User account(s) activated." message |
|  | 5. Huddle sends an email to user account owner to notify them that their account has been reactivated. |
| **Conclusion:** | Selected deactivated user account reactivated. | |
| **Post condition:** | All user accounts are displayed in the CMS for access by only the system Owner. Reactivated user accounts will be able to login into system | |

### 8.4.12 Use-case description for Create Project

|  |  |  |
| --- | --- | --- |
| **Use-Case ID:** | UC-12 | |
| **Use-Case Name:** | Create Project | |
| **Participating actor(s):** | Registered User | |
| **Description:** | Registered User to create project | |
| **Precondition:** | Huddle must be accessible | |
| System Owner must access CMS | |
| Must be a registered user | |
| **Typical course of events:** | **Actor Action** | **System Response** |
| 1. User to click on Create Project | 2. System to display the following fields to fill in: - Project Name: - Project duration: - Project start date: |
| 3. User to enter the fields and click Save or Cancel | 4a. If user clicks Save, system will save the entries and displays "Project saved successfully". System will display the dashboard.   4b. If user clicks Cancel, system will display the dashboard again. |
| **Conclusion:** | Project created | |
| **Post condition:** | From action item 3, if user clicks Save after entering the fields, the system will display recorded project details at the dashboard. | |

### 8.4.13 Use-case description for Update Project

|  |  |  |
| --- | --- | --- |
| **Use-Case ID:** | UC-13 | |
| **Use-Case Name:** | Update project | |
| **Participating actor(s):** | Project Owner | |
| **Description:** | Project Owner to update the project details with remarks | |
| **Precondition:** | Huddle must be accessible | |
| System Owner must access CMS | |
| Must be a project owner who owns at least 1 project | |
| Must be a registered user | |
| **Typical course of events:** | **Actor Action** | **System Response** |
| 1. User to select the project from the dashboard and click Update | 2. System to display the following fields from the project with existing value: - Project Name: - Project duration: - Project start date: |
| 3. User to make changes and click on Save or Cancel button | 4a. If user clicks Save, system will save the entries and displays "Project saved successfully". System will display the dashboard.   4b. If user clicks Cancel, system will display the dashboard again. |
| **Conclusion:** | Project updated | |
| **Post condition:** | From action item 3, if user clicks Save after amending the fields, the system will display recorded project details at the dashboard. | |

### 8.4.14 Use-case description for Delete Project

|  |  |  |
| --- | --- | --- |
| **Use-Case ID:** | UC-14 | |
| **Use-Case Name:** | Delete project | |
| **Participating actor(s):** | Project Owner | |
| **Description:** | Project Owner to update the project details with remarks | |
| **Precondition:** | Huddle must be accessible | |
| System Owner must access CMS | |
| Must be a project owner who owns at least 1 project | |
| Must be a registered user | |
| **Typical course of events:** | **Actor Action** | **System Response** |
| 1. User to select the project from the dashboard and click Delete | 2. System to display the confirmation on deletion for the selected project. |
| 3. User to click Delete or Cancel. | 4a. If user clicks Delete, system will delete the entries and displays "Project Deleted successfully". System will display the dashboard.   4b. If user clicks Cancel, system will display the dashboard again. |
| **Conclusion:** | Project will be deleted | |
| **Post condition:** | From action item 3, if user clicks deletes, the system will display recorded project details at the dashboard. | |

### 8.4.15 Use-case description for View Project

|  |  |  |
| --- | --- | --- |
| **Use-Case ID:** | UC-15 | |
| **Use-Case Name:** | View Project | |
| **Participating actor(s):** | Project Owner Project Member | |
| **Description:** | Project Owner to view project details | |
| **Precondition:** | Huddle must be accessible | |
| System Owner must access CMS | |
| Must be a registered user | |
| **Typical course of events:** | **Actor Action** | **System Response** |
| 1. User to click on the title of the project | 2. System to display the details for the selected project. |
| 3. User to select the follwing options: - Update - Delete - Cancel - Add Task | 4a. If user clicks Delete, Use Case 07 (UC07) will occur.   4b. If user clicks Cancel, system will display the dashboard again.   4c, If user clicks update, Use Case 06 (UC06) will occur.   4d. If user clicks Add Task, Use Case 09 (UC09) will occur |
| **Conclusion:** | Project displayed | |
| **Post condition:** | From action item 3, if user clicks Save after amending the fields, the system will display recorded project details at the dashboard. | |

### 8.4.16 Use-case description for Add Team Member

|  |  |  |
| --- | --- | --- |
| **Use-Case ID:** | UC-16 | |
| **Use-Case Name:** | Project Owner adds team members to project | |
| **Participating actor(s):** | Project Owner, team member | |
| **Description:** | An existing user who has already created a project adds team members into his/her project | |
| **Precondition:** | Huddle must be accessible | |
| User must have access to website | |
| User must have already registered for an account and account is in "verified" status | |
| User must have registered with a valid email account | |
| Account must not be deactivated | |
| **Typical course of events:** | **Actor Action** | **System Response** |
| 1. User accesses user dashboard |  |
| 2. User selects projects | 3. Huddle shows project list |
| 4. User clicks "New Project" button | 5. Huddle displays project creation form |
| 6. User inputs project details into project creation form and clicks "Create" | 7. Huddle saves information in project creation form into database as a new unique project. If unsuccesful will display "Project not created. Please try again." message. If successful will display "Project created successfully." message and will bring user to add team member page. |
|  | 8. Huddle updates user status to Project Owner for particular project |
| 9. Project Owner searches for team members by inputting email address of team members into search bar and clicks "Search" | 10. Huddle checks if email in query matches any records in database. If no match will display "No such user found. Please try again." message. If there is a match, will display user summarized details (eg avatar image, name, email) |
| 11. Project Owner selects team member profile summary and clicks add | 12. Huddle assigns team member to project and updates status to Team Member to particlar project. If unsuccessful will display "Team member not added to project. Please try again." message. If successful will display "Team member successfully added." message. |
|  | 13. Huddle sends an email notification to Team Member |
| 14. Project Owner clicks on done | 15. Huddle updates status and links of user accounts into database |
|  | 16. Huddle updates project list |
|  | 17. Huddle displays user dashboard |
| **Conclusion:** | Users create project and status changes to Project Owner. Team members are added to the project and have status changed to Team Member. | |
| **Post condition:** | Project Owners and Team Members have access to project and its details, each with different authorisation levels. | |

### 8.4.17 Use-case description for View Team Member

|  |  |  |
| --- | --- | --- |
| **Use-Case ID:** | UC-17 | |
| **Use-Case Name:** | View team members | |
| **Participating actor(s):** | Project Owner, team member | |
| **Description:** | Existing user to view team members added in the project | |
| **Precondition:** | Huddle must be accessible | |
| There must be a project |  |
| Must be from UC15 |  |
| User must have access to website | |
| User must have already registered for an account and account is in "verified" status | |
| User must have registered with a valid email account | |
| Account must not be deactivated | |
| **Typical course of events:** | **Actor Action** | **System Response** |
| 1. User to click View Team Members | 2. System to display team members' name, allocated task and due date |
| **Alternative event** |  | 3. System will show message "No team members allocated" and displays a button to perform "Add member" use case |
| **Conclusion:** | User be able to view the project team members in the project | |
| **Post condition:** | Team members have been displayed. | |

### 8.4.18 Use-case description for Remove Team Member

|  |  |  |
| --- | --- | --- |
| **Use-Case ID:** | UC-18 | |
| **Use-Case Name:** | Remove team members | |
| **Participating actor(s):** | Project Owner | |
| **Description:** | Existing user to remove team members added in the project | |
| **Precondition:** | Huddle must be accessible | |
| There must be a project | |
| There must be at least 1 member added in project | |
| Must be from UC08 | |
| User must have access to website | |
| User must have already registered for an account and account is in "verified" status | |
| User must have registered with a valid email account | |
| Account must not be deactivated | |
| **Typical course of events:** | **Actor Action** | **System Response** |
| 1. User to click "X" button to remove user | 2. System to display message " Are you sure to remove member? Assigned task will be unassigned" and "Yes" and "No" option |
| 3. User selects "Yes" | 4. System displays "Member removed successful" and record will be deleted. |
| **Alternative event** | 5. From event 2, user clicks "No" | 6. System will return to view projects. |
| **Conclusion:** | User will be deleted | |
| **Post condition:** | Team member removed | |

### 8.4.19 Use-case description for Create Activity

|  |  |  |
| --- | --- | --- |
| **Use-Case ID:** | UC-19 | |
| **Use-Case Name:** | Create Activity | |
| **Participating actor(s):** | Project owner | |
| **Description:** | Registered User to create project | |
| **Precondition:** | Huddle must be accessible | |
| System Owner must access CMS | |
| Must be a project owner who owns at least 1 project | |
| Must be a registered user | |
| **Typical course of events:** | **Actor Action** | **System Response** |
| 1. User to click on Create Activity | 2. System to display the following fields to fill in: - Activity Name |
| 3. User to enter the fields and click Save or Cancel | 4a. If user clicks Save, system will save the entries and displays "Activity name saved successfully". System will display the dashboard.   4b. If user clicks Cancel, system will display the dashboard again. |
| **Conclusion:** | Activity created | |
| **Post condition:** | From action item 3, if user clicks Save after entering the fields, the system will display recorded project details at the dashboard. | |

### 8.4.20 Use-case description for Update Activity

|  |  |  |
| --- | --- | --- |
| **Use-Case ID:** | UC-20 | |
| **Use-Case Name:** | Update Activity | |
| **Participating actor(s):** | Project Owner | |
| **Description:** | Project Owner to update the activity | |
| **Precondition:** | Huddle must be accessible | |
| System Owner must access CMS | |
| Must be a project owner who owns at least 1 project | |
| Must be a registered user | |
| **Typical course of events:** | **Actor Action** | **System Response** |
| 1. User to click on the activity name | 2. System to display the following fields from the project with existing value: |
| 3. User to make changes and click on Save or Cancel button | 4a. If user clicks Save, system will save the entries and displays "Activity name saved successfully". System will display the dashboard.   4b. If user clicks Cancel, system will display the dashboard again. |
| **Conclusion:** | Activity name updated | |
| **Post condition:** | From action item 3, if user clicks Save after amending the fields, the system will display recorded project details at the dashboard. | |

### 8.4.21 Use-case description for Delete Activity

|  |  |  |
| --- | --- | --- |
| **Use-Case ID:** | UC-21 | |
| **Use-Case Name:** | Delete Activity | |
| **Participating actor(s):** | Project Owner | |
| **Description:** | Project Owner to delete activity | |
| **Precondition:** | Huddle must be accessible | |
| System Owner must access CMS | |
| Must be a project owner who owns at least 1 project | |
| Must be a registered user | |
| **Typical course of events:** | **Actor Action** | **System Response** |
| 1. User to select the Activity from the project and click Delete | 2. System to display "Are you sure? The following tasks will be removed as well". |
| 3. User to click Delete or Cancel. | 4a. If user clicks Delete, system will delete the entries and displays "Activity Deleted successfully". System will display the dashboard.   4b. If user clicks Cancel, system will display the dashboard again. |
| **Conclusion:** | Activity removed successfully | |
| **Post condition:** | From action item 3, if user clicks Save after amending the fields, the system will display recorded project details at the dashboard. | |

### 8.4.22 Use-case description for Create Task

|  |  |  |
| --- | --- | --- |
| **Use-Case ID:** | UC-22 | |
| **Use-Case Name:** | Create Task | |
| **Participating actor(s):** | Project Owner | |
| **Description:** | Project Owner to add task for project | |
| **Precondition:** | Huddle must be accessible | |
| UC19 must be completed | |
| Must be from UC12 | |
| Must be a registered user | |
| **Typical course of events:** | **Actor Action** | **System Response** |
| 1. User to Click Add Task | 2. System to display the details for user input as follows: - Task name - Task type - Task duration - Task start date - Task member (Select from drop box) |
| 3. User to enter the fields and click Save or Cancel | 4a. If user clicks Save, system will save the entries and displays "Project tasks saved successfully". System will display the dashboard.   4b. If user clicks Cancel, system will display the dashboard again. |
| **Conclusion:** | Task inserted. | |
| **Post condition:** | From action item 3, if user clicks Save after amending the fields, the system will display recorded project details at the dashboard. | |

### 8.4.23 Use-case description for Update Task

|  |  |  |
| --- | --- | --- |
| **Use-Case ID:** | UC-23 | |
| **Use-Case Name:** | Update Task | |
| **Participating actor(s):** | Project Owner Project Member | |
| **Description:** | Project Owner or Project Member to update task | |
| **Precondition:** | Huddle must be accessible | |
| Must be from UC 09 | |
| Must be a registered user | |
| **Typical course of events:** | **Actor Action** | **System Response** |
| 1. User to select task and click "Update" | 2. System to display the following task details owned by the project member: - Description of task  - Remarks - Date to be completed - T-minus remaining days.  - Upload attachment button - Comments on attachment - Remove attachment button |
| 3. User to click upload file | 4. If user clicks Upload, system will display the file explorer for user to upload file. |
| 5. User to select the correct directory and click on the files. User to click "Ok" | 6. System will then display the uploading status.   6a. If upload is successful, system will display "Upload successful".  6b. If upload failed, system will display "Upload unsuccessful, please try again." |
| **Conclusion:** | Task updated by project member. | |
| **Post condition:** | From action item 5, if user clicks Save after amending the fields, the system will display recorded project details at the dashboard. | |

### 8.4.24 Use-case description for Delete Task

|  |  |  |
| --- | --- | --- |
| **Use-Case ID:** | UC-24 | |
| **Use-Case Name:** | Delete task | |
| **Participating actor(s):** | Project Owner | |
| **Description:** | Project Owner to remove task from a project | |
| **Precondition:** | Huddle must be accessible | |
| System Owner must access CMS | |
| Project must have at least 1 task added | |
| Must be from UC12 | |
| Must be a project owner who owns at least 1 project | |
| Must be a registered user | |
| **Typical course of events:** | **Actor Action** | **System Response** |
| 1. User to select task and click "Delete" | 2. System to display "Are you sure you want to delete the task?" |
| 3. User to click Delete or Cancel. | 4a. If user clicks Delete, system will delete the entries and displays "Task Deleted successfully". System will display the dashboard.   4b. If user clicks Cancel, system will display the dashboard again. |
| **Conclusion:** | Task deleted | |
| **Post condition:** | From action item 3, if user clicks Save after amending the fields, the system will display recorded project details at the dashboard. | |

### 8.4.25 Use-case description for View Task

|  |  |  |
| --- | --- | --- |
| **Use-Case ID:** | UC-25 | |
| **Use-Case Name:** | View Task | |
| **Participating actor(s):** | System Owner  Project Owner Project Member | |
| **Description:** | Project Owner and Member to view the project tasks | |
| **Precondition:** | Huddle must be accessible | |
| System Owner must access CMS | |
| Project must have at least 1 task added | |
| Must be from UC09 | |
| Must be a project owner who owns at least 1 project | |
| Must be a registered user | |
| **Typical course of events:** | **Actor Action** | **System Response** |
| 1. User to click on the title of the project | 2. System to display the details for the selected project. |
| 3. User to select the task required to be displayed | 4. System will display the task details |
| **Conclusion:** | Task details displayed | |
| **Post condition:** | Task will be displayed | |

### 8.4.26 Use-case description for Upload Attachment

|  |  |  |
| --- | --- | --- |
| **Use-Case ID:** | UC-26 | |
| **Use-Case Name:** | Upload attachment | |
| **Participating actor(s):** | Project Owner Project Member | |
| **Description:** | Project Owner and Member to upload attachments. | |
| **Precondition:** | Huddle must be accessible | |
| System Owner must access CMS | |
| Project must have at least 1 task added | |
| Must be from UC14 | |
| Must be a project owner who owns at least 1 project | |
| Must be a registered user | |
| **Typical course of events:** | **Actor Action** | **System Response** |
| 1. User to click upload file | 2. If user clicks Upload, system will display the file explorer for user to upload file. |
| 3. User to select the correct directory and click on the files. User to click "Ok" | 3. System will then display the uploading status.   3a. If upload is successful, system will display "Upload successful".  3b. If upload failed, system will display "Upload unsuccessful, please try again." |
| **Conclusion:** | Attachment uploaded | |
| **Post condition:** | System to display the dashboard with the list of projects | |

### 8.4.27 Use-case description for Remove Attachment

|  |  |  |
| --- | --- | --- |
| **Use-Case ID:** | UC-27 | |
| **Use-Case Name:** | Delete attachment | |
| **Participating actor(s):** | Project Owner Project Member | |
| **Description:** | Project Owner and Member to remove attachment from task | |
| **Precondition:** | Huddle must be accessible | |
| System Owner must access CMS | |
| Project must have at least 1 task added | |
| Must be from UC12 | |
| Must be a project owner who owns at least 1 project | |
| Must be a registered user | |
| **Typical course of events:** | **Actor Action** | **System Response** |
| 1. User to click remove attachment | 2. System will display "Are you sure to delete the attachment?" |
| 3. User click "Yes" | 3. System will then display the uploading status.   3a. If removal is successful, system will display "Remove successful".  3b. If upload failed, system will display "Remove unsuccessful, please try again." |
| **Alternative event** | 4. From 2, User clicks "No" | 5. System will go back to UC12 |
| **Conclusion:** | Attachment uploaded | |
| **Post condition:** | System to display the dashboard with the list of projects | |

### 8.4.28 Use-case description for Download Attachment

|  |  |  |
| --- | --- | --- |
| **Use-Case ID:** | UC-28 | |
| **Use-Case Name:** | Download attachment | |
| **Participating actor(s):** | Project Owner Project Member | |
| **Description:** | Project Owner and Member to remove attachment from task | |
| **Precondition:** | Huddle must be accessible | |
| System Owner must access CMS | |
| Project must have at least 1 task added | |
| Must be from UC22 | |
| Must be a project owner who owns at least 1 project | |
| Must be a registered user | |
| **Typical course of events:** | **Actor Action** | **System Response** |
| 1. User to click download attachment | 2. System will start download attachment |
| **Alternative event** |  | 5. If there are any connectivity or issues downloading, system will display "Error downloading, please try again. |
| **Conclusion:** | Attachment downloaded | |
| **Post condition:** | Attachment downloaded | |

### 8.4.29 Use-case description for Email User

|  |  |  |
| --- | --- | --- |
| **Use-Case ID:** | UC-29 | |
| **Use-Case Name:** | Email User | |
| **Participating actor(s):** | Owner Project Owner Project Member | |
| **Description:** | A function for actors to send email to users in Huddle system | |
| **Precondition:** | Huddle must be accessible | |
| User must have access to website | |
| Must be from UC08 | |
| User must have already registered for an account | |
| User must have registered with a valid email account | |
| Account must not be deactivated | |
| **Typical course of events:** | **Actor Action** | **System Response** |
| 1. User to click on member's name | 2. System will open a local mail client with the TO field filled with recipient's email address. |
| **Conclusion:** | User will send email to other users | |
| **Post condition:** | System will return to dashboard | |

### 8.4.30 Use-case description for Project Search

|  |  |  |
| --- | --- | --- |
| **Use-Case ID:** | UC-30 | |
| **Use-Case Name:** | Search projects | |
| **Participating actor(s):** | Owner Project Owner Project Member | |
| **Description:** | A function for actors to search for projects assigned in Huddle system | |
| **Precondition:** | Huddle must be accessible | |
| User must have access to website | |
| User must have already registered for an account | |
| User must have registered with a valid email account | |
| Account must not be deactivated | |
| **Typical course of events:** | **Actor Action** | **System Response** |
| 1. User to click on search projects | 2. System will display the search field for user input |
| 3. User to enter keywords | 4. System will display the details with the approximate match with keyword |
| **Conclusion:** | Results displayed to users | |
| **Post condition:** | Results displayed to users | |

### 8.4.31 Use-case description for Assign Project

|  |  |  |
| --- | --- | --- |
| **Use-Case ID:** | UC-31 | |
| **Use-Case Name:** | Assign project | |
| **Participating actor(s):** | System Owner | |
| **Description:** | System Owner to assign Project Owner to existing projects | |
| **Precondition:** | Huddle must be accessible | |
| There must be at least 1 project owner | |
| System administrator must access CMS | |
| There must be at least 1 project | |
| **Typical course of events:** | **Actor Action** | **System Response** |
| 1. System owner to click on "Assign project owner" | 2. System will then show dropbox to select project owner |
| 3. System owner to select Project Owner and click save | 4. System will display "Assignment successful" and save the new record |
| **Alternative course of events** | 5 From item 3, System owner clicks "Cancel" | 6. System will display "View Profile" page for System Owner |
| **Conclusion:** | Project Owner has been assigned | |
| **Post condition:** | Project has a new Project Owner | |

# 8.5 Models for processes in the system

### 8.5.1 Context Diagram

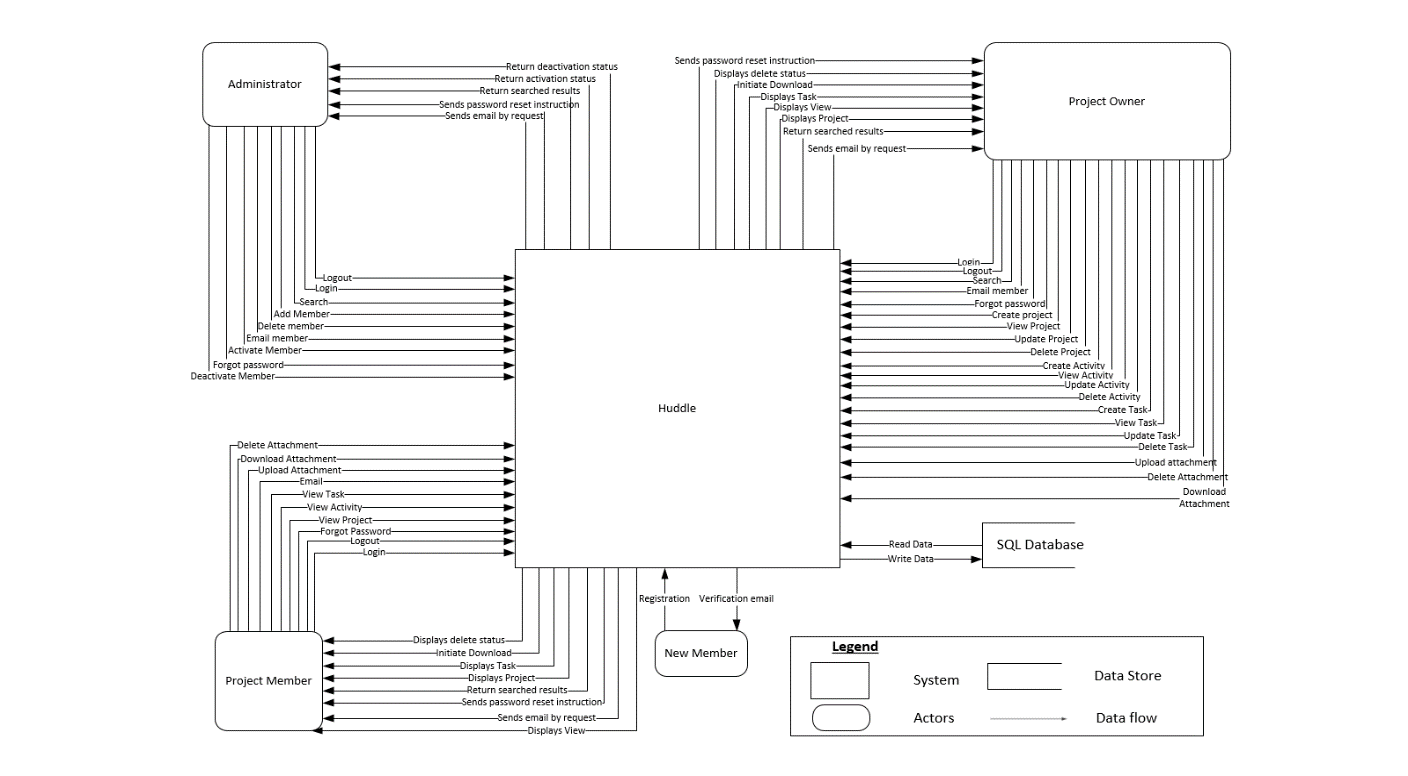


Figure 6 Context Data Flow Diagram

*Figure 6* depicts that there are three different types of processes that would interact with the system, and they each would have specific roles in the development of the system. All three processes has their specific roles and none of their actions would overlap.

# 8.6 Models for data in the system

### 8.6.1 Level 0 Entity-Relationship Diagram

Figure 7 Entity Relationship Diagram

*Figure 7* shows all the entities in the system and their interactions. The user entity can either be a system administrator; or a project owner, or a team member. A project may only have one project owner, but can have many team members. Projects are divided into activities, and each activity can be further divided into tasks. An example will be the project is the Huddle website, one activity as homepage, and tasks include update content for the homepage, page web design etc.

# 8.7 CRUD Matrix

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Task | System Administrator | User | Project Owner | Team Member |
| Add New User | CR |  |  |  |
| Activate User | RU |  |  |  |
| Deactivate User | RU |  |  |  |
| Delete User | RD |  |  |  |
| Assign User as Project Owner | CUD |  |  |  |
| Register Account |  | C |  |  |
| User Profile | CRUD | CRU | CRU | CRU |
| Create, View, Edit Project | CRUD |  | CRUD | R |
| Create, View, Edit Activity | CRUD |  | CRUD | R |
| Create, View, Edit Task | CRUD |  | CRUD | RU |
| Add Team Member | CRUD |  | RU |  |
| Documents | CRUD |  | CRUD | CRUD |
| Legend : C = Create, R = Read, U = Update, D = Delete | | | | |

|  |  |
| --- | --- |
| Actors | Brief Description |
| System Administrator | Able to create user account  Able to deactivate and reactivate the account  Able to delete user account from system  Able to assign user as Project Owner  Able to create, edit and view user profile  Able to create, edit and view Project  Able to create, edit and view Project Activity  Able to create, edit and view Project Task  Able to create, edit and view Project Documents  Able to create, edit and view Team Member |
| User | Able to register for an account  Able to create, edit and view user profile |
| Project Owner | Able to create, edit and view own profile  Able to create, edit and view Project  Able to create, edit and view Project Activity  Able to create, edit and view Project Task  Able to create, edit and view Project Documents  Able to create, edit and view Team Member |
| Team Member | Able to create, edit and view own profile  Able to view Project  Able to view Project Activity  Able to edit and view Project Task  Able to create, edit and view Project Documents |

Above tables show a CRUD Matrix, which show the different authorisation levels each actor has in the Creating, Reading, Updating, and Deleting of data in the System. This will be in line with the business rules for authorisation levels of different users, and will allow for some control over who has the authorisation to do sensitive actions such as the deleting of data in the system.

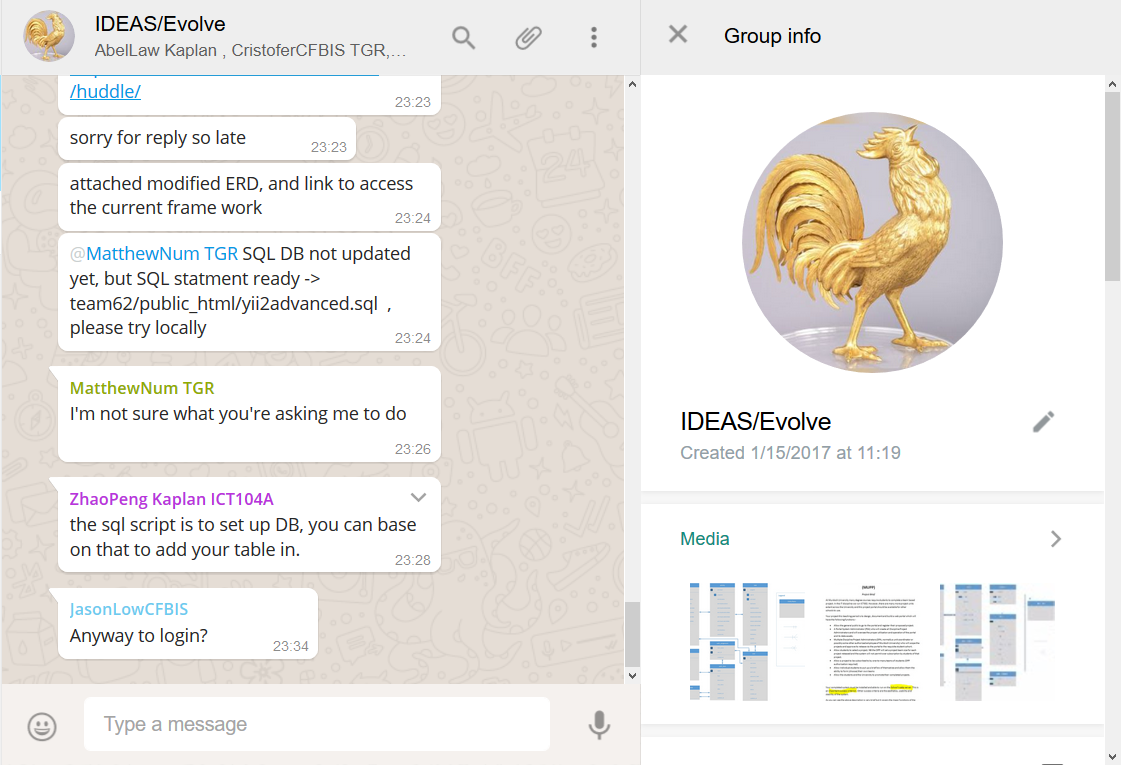
# **9.0 Working with External Party**

This section documents the working with an external party (Team EVOLVE) towards the submission of one deliverable. It highlights the communication shared, as well as any problems that surfaced, and the actions done to rectify such problems.

## 9.1 Chat Group Communication Channel

Main communication chat group: Team IDEAS EVOLVE WhatsApp chat group created on 15th January 2017 by Nicol Loh Chuen (Sole group admin)

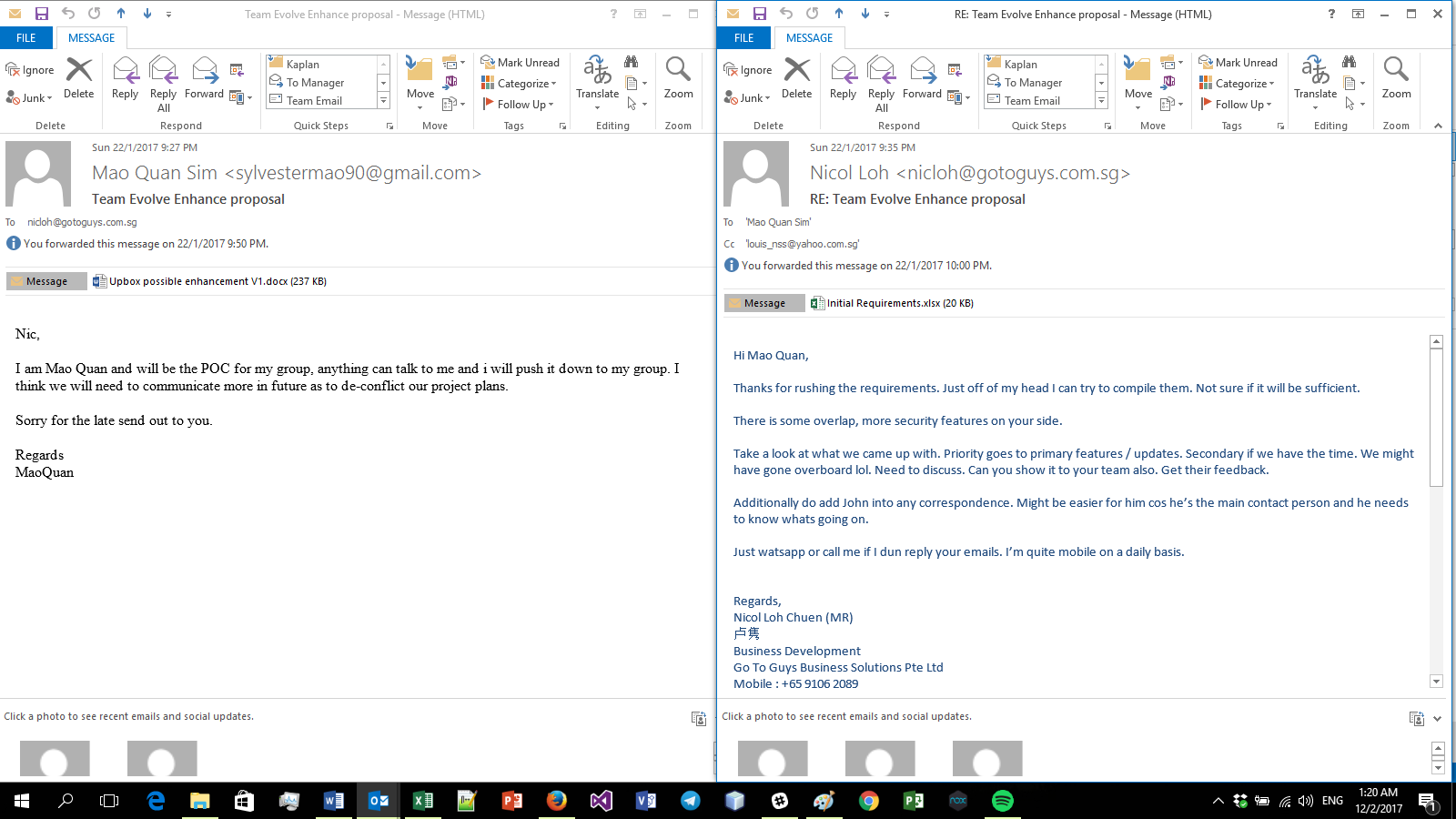
Members: All team members of Teams IDEAS and EVOLVE



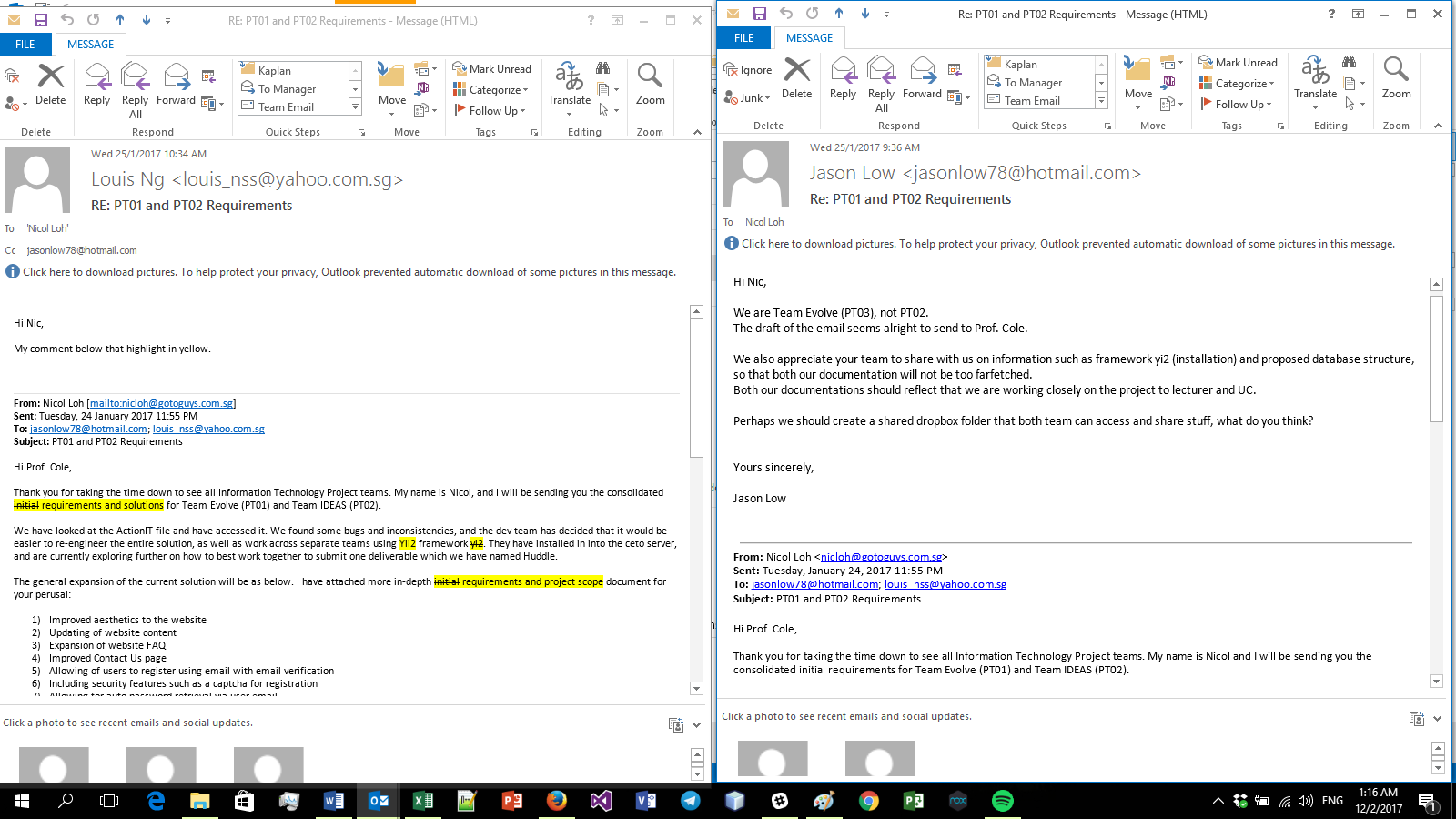
## 9.2 Important Email Correspondences Between Teams:

Dated: 22nd January 2017

Subject: Compiling of both teams’ requirements for email to client



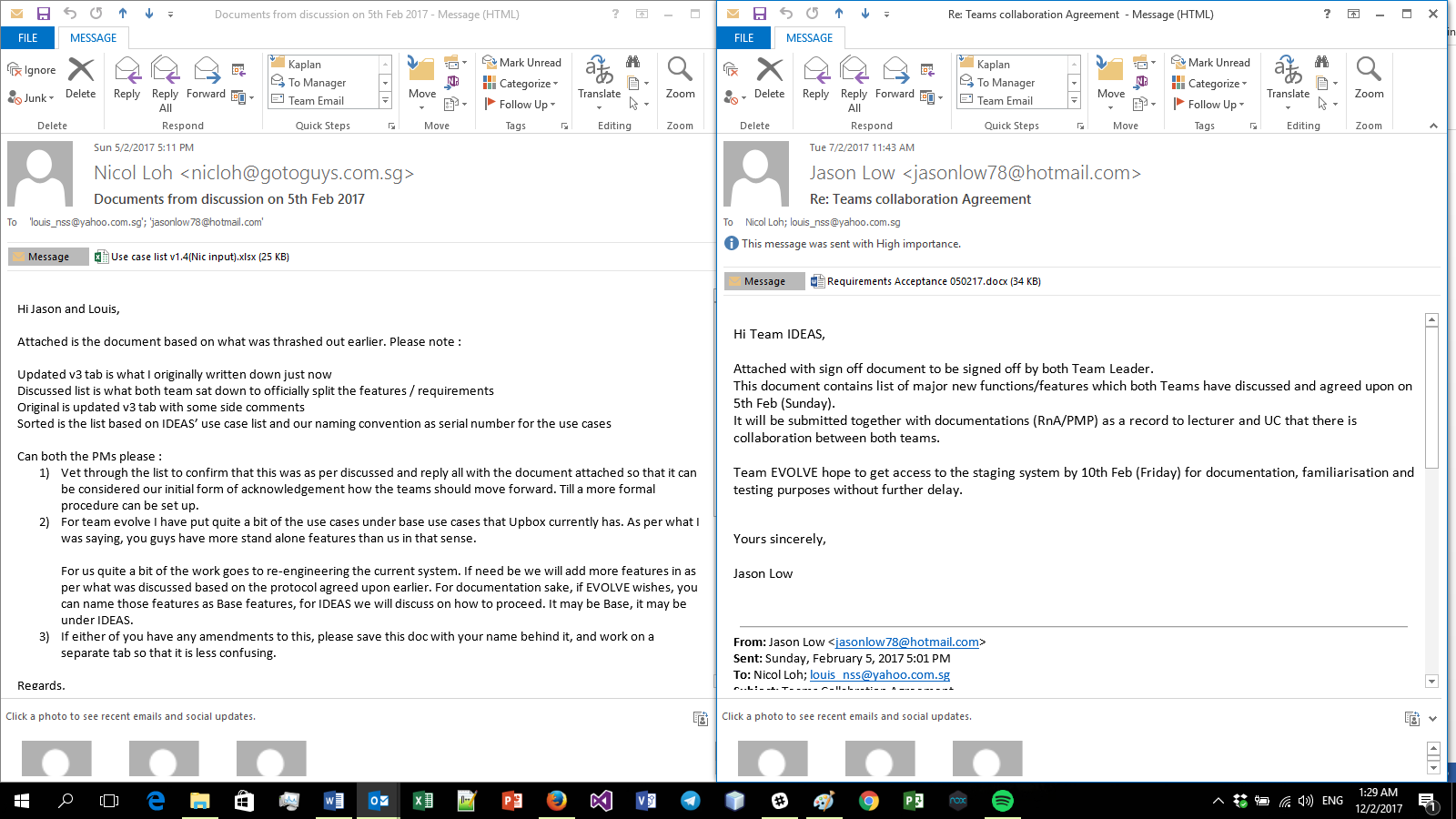
Dated: 24th to 25th January 2017

Subject: Confirmation of combined scope and email to be approved by both teams’ project managers prior to sending to client

Dated :

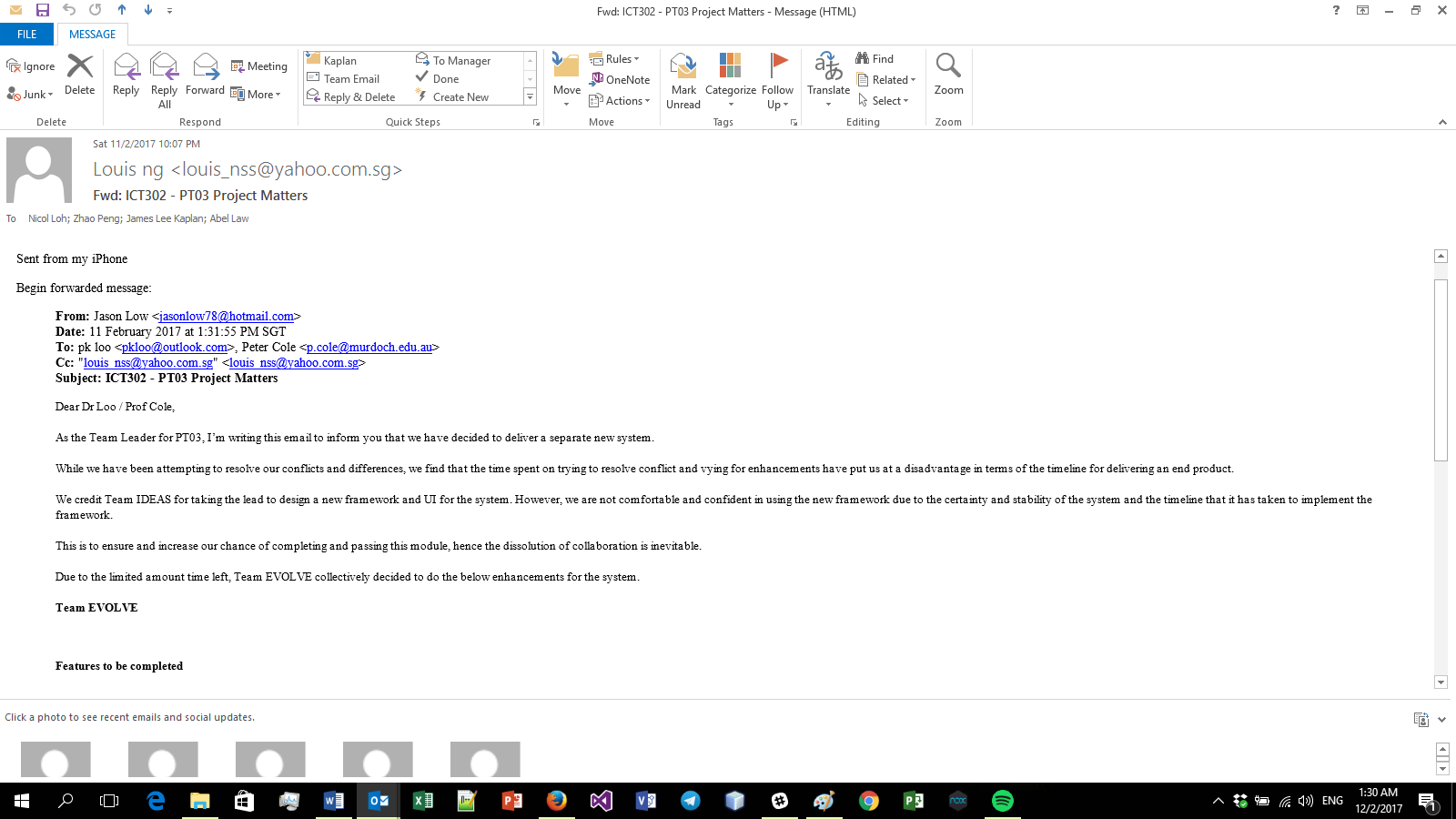
Dated: 5th to 7th February 2017

Subject: Plan for integration of both teams projects



Dated: 11th February 2017

Subject: Official notice of Team Evolve’s decision to split from Team IDEAS



## 9.3 Challenge Faced

The key challenge that Team IDEAS faced is when the external team first initiated a split from working together to delivering one deliverable due to doubts and/or unfamiliarity with the chosen Yii2 framework. The solution that we came up with was that Team IDEAS offered to integrate Team EVOLVE’s features into the built framework so that both teams are able to submit one deliverable. The result of this was the official splitting of requirements by both sides teams to remove ambiguity, as well as the coming up of some conditions for the assisting of the integration process which was agreed upon by both teams. This led to greater communication between the teams and was agreed upon to be the start for procedures to be in place that are fair for both teams.

# **10.0 Conclusion**

This Requirements and Analysis Document shows the skeletal structure of Huddle, and gives sufficient detail to non-technical people as well as to the developers on what the vision for the system is. It is through this document that Team IDEAS will use as a basis to create the Project Management Tool for usage across all industries.

Huddle is a simple and easy to use tool with a short and shallow learning curve, and yet conceptualised in a simple elegant manner, which allows for multiple tiers of projects interconnected via project owners and team member relationships. It helps with the removal of ambiguity, increases communication, decreases waiting time and increases productivity, with automation making it a scalable system.