Milestone 2

Team 5

Bohlokoa Tilo (1852529)

Chico Vicente (1844374)

Bolekwa Mboniswa (1397266)

Xlolani Magoloza (1412049)

Rahube Seabi (1834316)

# Action Points

|  |  |
| --- | --- |
| **Feedback** | **Action Point** |
| The purpose of our milestone was lacking in terms of actively pushing the client towards sign off | Mention something about the document serving as a sign-off document between us (Kelello) and the organization (#fundMe). |
| The problem statement came across as a solution and not as a problem due to a lack of understanding of what it is | Reevaluate the study case with a fine-tooth comb to gain better understanding of what the problem actually is. |
| The Class Diagram had two erroneous relationships between entities | The Application to Applicant relationship should be 1..1 as we can’t have an Applicant without an Application  The same can be said for the relationship between Donor and Admin |
| The Case Set was not clearly stated to define the “read” methods | We should use the word Read instead of Retrieve as this does not clearly express the method. |
| The team failed to distinguish between functional and non-functional requirements | We as team need to reevaluate our understanding of functional and non-functional requirements by using different examples/scenarios to practice how to identify and differentiate between the two kinds of requirements. |

# 2. Purpose

This document is the second iteration in mapping and showing our current understanding of your organization’s domain and presenting the framework of a system that aims help combat your current problem. Its purpose is to act as a form of contractual agreement between Kelello and the organisation, #fundMe. The document certifies that the organisation, #fundMe has read and reviewed the document and are satisfied with the information presented within the document, thus far in the project; and if not, the issues are to be communicated back to us (Kelello) for further modification. It therefore serves as a sign-off document between the two abovementioned parties to supply each with a form of legal protection.

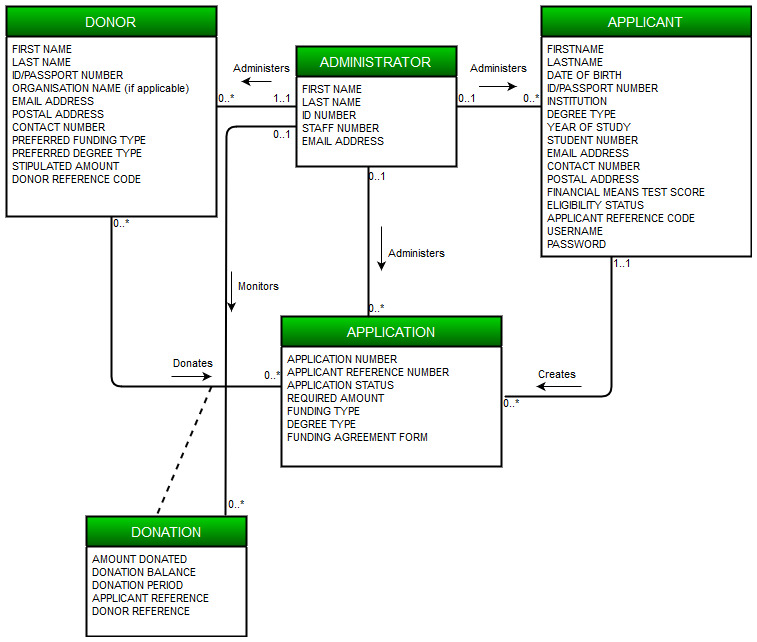
# Executive Summary

A large portion of this milestone is reflecting, and iterating based off the shortcomings of Milestone 1. The first part of this was addressing the 5 most important feedback points and developing them into action points. These action points helped create consistency across our models and show improved understanding of the client domain, an example of this was how we shifted our understanding of what constitutes a functional requirement in this context.

We then proceeded to revise the Class Diagram, Use Case Set and Diagram according to these action points. Notable changes include changing the type of relationships in the Class Diagram and making the wording we use in the Use Case Set more consistent with UML standards.

We then developed some of the use cases into Fully Dressed Use Cases. The Fully Dressed Use Cases contains: a brief description on the task the use case is supposed to perform, a list of the actors on the system, a list of all the stakeholders in the use case and their interests, pre and post conditions, and most importantly the flow of activities which is central to developing the SSD. The Fully Dressed Use Cases are each supported by their respective Use Case Diagram and SSD, the latter which is mainly to model the communication between the system and involved actors. The isolated Use Case is mostly used to model the invocation of other Use Cases.

# 4. Revised Class Diagram for the #fundMe system

The following is a class diagram which serves as a representation of the structure of the system’s database. It also shows the relationships among the tables of the database, the relationships of which we elicited from the given business rules. In this case, the tables are Applicant, Application, Donor, Donation and Administrator, as can be seen in the diagram below.

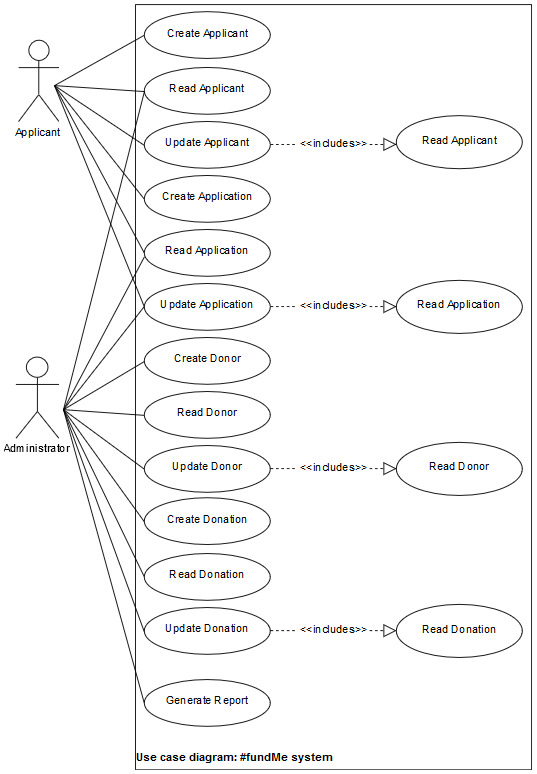
# 5. Revised Use case set for the #fundMe system

The following table is an illustration of a use case set, which essentially serves as a framework of the events that are expected to take place within the #fundMe system. Applicant, Application, Donor, Donation and Administrator are the tables within the system’s database as seen in the [Class Diagram](#ERD) above. The Create, Read and Update are the core operations that are to trigger the attention of the system.

|  |  |  |  |
| --- | --- | --- | --- |
|  | Use Case | Description | Used use cases |
| APPLICANT | | | |
|  | Create Applicant | Registration is required for first time applicants. |  |
| Read Applicant | Allows applicant to view applicant details. |  |
| Update Applicant | Allows applicant to update applicant details. | Read applicant |
| APPLICATION | | | |
|  | Create Application | Allows user to apply for required funding. |  |
| Read Application | Allows user to view application from the database. |  |
| Update Application | Allows user to update application details. | Read application |
| DONOR | | | |
|  | Create Donor | Each new donor must be created by an administrator. |  |
| Read Donor | Allows admin to view a donor’s details. |  |
| Update Donor | Allows admin to update a donor’s details. | Read Donor |
| DONATION | | | |
|  | Create Donation | A new donation is created when a match has been made between applicant and donor. |  |
| Read Donation | Allows user to view donation details. |  |
| Update Donation | Allows user to update donation details. | Read Donation |
| ADMINISTATOR | | | |
|  | Create Administrator | Creates new administrator. |  |
| Read Administrator | Reads administrator’s details. |  |
| Update Administrator | Updates administrator’s details. | Read Administrator |
| REPORT | | | |
|  | Generate report | | |

# 6. Revised Use Case Diagram for #fundMe system

The use case diagram illustrates the interaction of the users of the system, Applicant and Administrator, with the system, as well as which operation, according to the use case set above, concerns which users. In this particular case both the Applicant and the Administrator are primary actors of the system.

The donor plays no role in the system and thus they do not appear as an actor of the system in the diagram.

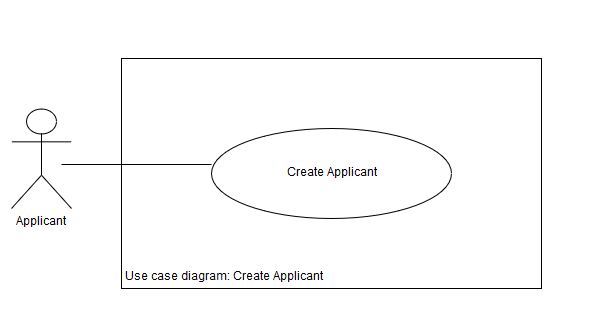
# 7.1.1 Fully-dressed use case description for Create Applicant

The below fully-dressed use case description provides detailed information on create applicant. It provides necessary steps on process on how the actor interacts with the system, what must be true/false for the activities to occur.

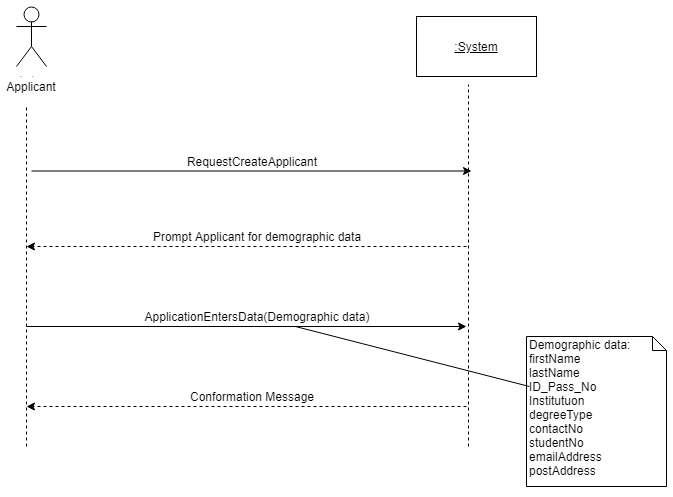
|  |  |  |
| --- | --- | --- |
| **Use case name:** | Create Applicant | |
| **Scope:** | #FundMe System | |
| **Triggering Event:** | Applicant requests to create applicant | |
| **Brief Description:** | When a student wants to be an applicant in the system they can request to create applicant, the system will then allow them to enter their demographic data, the system will then capture the demographic data into the APPLICANT table. The system will then generate and send a confirmation message with login details to the applicant. | |
| **Actors:** | Applicant (Primary) | |
| **Related Use Cases:** | N/A | |
| **Stakeholders and interests:** | Applicant: Desires to be captured in the system thus records their Data  Administrator: Desires to keep track of new applicants | |
| **Pre-conditions:** | N/A | |
| **Post Conditions:** | Applicant recorded in the #FundMe Database  Message sent to applicant to confirming data has been recorded as an Applicant to #FundMe | |
| **Flow of Activities** | Actor | System |
| 1.Applicant requests to Create Applicant  2.Applicant Enters the required Data | 1.1Prompts applicant to enter demographic data  2.1 Records the data into the APPLICANT table  2.3 Sets applicant funding eligibility status to “Red”  2.4 Generates login details  2.5 Sends a confirmation message to Applicant, containing login details and instructions to upload the necessary documentation |

# 7.1.2 Use case diagram for Create Applicant

The following use case diagram is a representation of the interaction between the process of create applicant and the applicant. Occurs when a user is a first time applicant.



# 7.1.3 System sequence diagram for Create Applicant

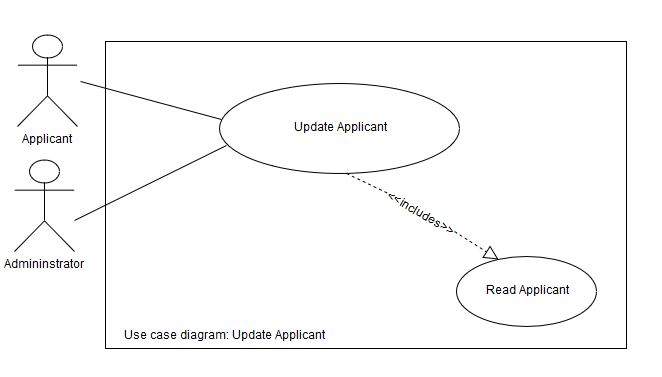
This diagram maps the communication between the system and the applicant during the Create Applicant process.

# 7.2.1 Fully-dressed use case description for Update Applicant

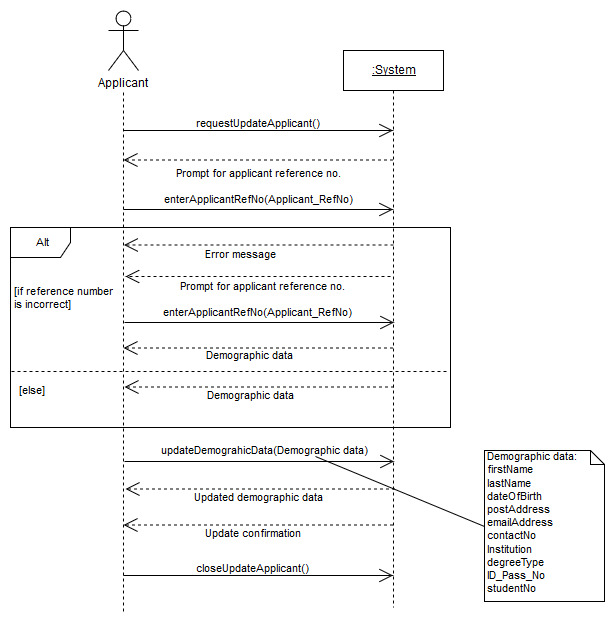
The fully-dressed use case description below gives a more detailed elucidation of the Update Applicant use case/process. It sheds light upon the steps necessary to perform the process successfully and exceptions/alternatives in the case of a deviation from the main path, as well as what must be true before and after the process takes place.

|  |  |  |
| --- | --- | --- |
| **Use case name:** | Update Applicant | |
| **Scope:** | #fundMe System | |
| **Triggering Event:** | Applicant requests to update applicant | |
| **Brief Description:** | When an Applicant wishes to update their demographic data, they can request to update applicant. The system will enable the applicant to make changes to applicant. The Applicant then makes the desired changes and saves those changes. The system then makes record of these changes and captures them into the APPLICANT data structure. After which it displays the updated version of the demographic data. The system then sends a confirmation message to the Applicant, pertaining to the changes made. | |
| **Actor(s):** | Applicant(Primary) | |
| **Related use case(s):** | Read Applicant | |
| **Stakeholder and interests:** | Applicant: Desires to update their details efficiently and with ease.  Administrator: Desires to always have up-to-date information about the applicant to avoid providing inaccurate information to donors and generating reports based off out-dated data.  The Organisation: Desires to be provided with accurate and up-to-date reports and information for decision making with regards to the impact of #fundMe. | |
| **Pre-conditions:** | The applicant’s record must already exists within the APPLICANT table in the #fundMe database. | |
| **Post-conditions:** | Updated demographic data are recorded into the APPLICANT table.  Generated confirmation message detailing the updates made to the Applicant’s demographic data.  Confirmation message detailing the updates made to the Applicant’s demographic data is sent to the Applicant. | |
| **Flow of activities:** | Actor | System |
| 1. Applicant requests to update applicant 2. Applicant enters applicant reference number 3. Applicant updates demographic data 4. Applicant saves updates   5.Appilcant closes update  applicant | * 1. Prompts for applicant   reference number   * 1. Verifies applicant reference number   2. Uses applicant reference number to search for Applicant’s record in the database   3. Invokes the Read Applicant use case   4. Displays Applicant’s demographic data   5. Records updates   6. Stores updates in APPLICANT table   7. Displays updated demographic data   8. Sends a confirmation message to the Applicant, pertaining to the updates made. |
| **Extensions:** | 1. Applicant enters   incorrect applicant reference number | A.1 Verifies applicant  reference number  A.2 Displays error message  A.3 Returns to [step 1.1](#Beginning) of the  flow of activities |

# 7.2.2 Use case diagram for Update Applicant

The following use case diagram is a representation of the interaction between the process of update applicant and the applicant. It also shows the related process, read applicant, which is necessary for update applicant to take place.

# 7.2.3 System sequence Diagram for Update Applicant

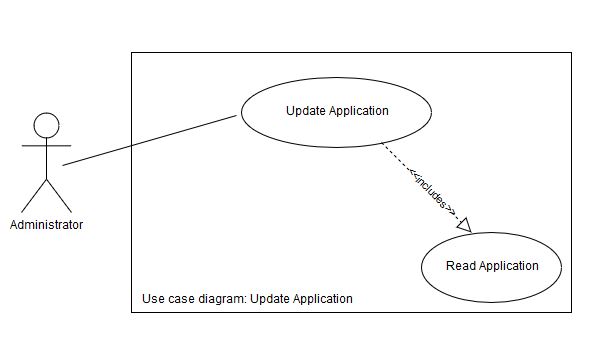
The following sequence diagram represents the order in which communication takes place between the applicant and the system during the update applicant process.

# 8.1 Fully dressed use case diagram for Update Application

Our team chose to do an Update Application fully dressed use case description for its complexity compared to the other use case descriptions and also how it expands our knowledge of fully dressed use case descriptions and their relevance to the project.

|  |  |  |
| --- | --- | --- |
| **Use Case Name:** | Update Application | |
| **Scope:** | #fundMe system | |
| **Triggering Event:** | When an Applicant or Administrator selects the option to edit an existing application | |
| **Brief Description:** | The Applicant or the Administrator chooses the option on the system to update an application. The Applicant/Administrator then enters the application reference number, which prompts the system to read the application record from the database.  **Process for Applicant:**  The system displays the application and enables the Applicant to update their funding type, degree type and/or required amount.  **Process for Administrator:**  The system displays the application and enables the Administrator to add the agreement form once a donor has been matched with an application and change the status of the application to ”Funded”.  The system captures the updates and stores them in the APPLICATION table, it then displays the updated version of the application. The system generates and sends a conformation message to the Applicant and Administrator pertaining the details of the update. | |
| **Actor(s):** | Applicant (Primary)  Administrator (Primary) | |
| **Related Use Cases:** | Read Applicant | |
| **Stakeholders and Interests:** | Applicant: Wants to update their funding type, degree type and/or required funding amount with ease  Administrator: Wants to upload the agreement form and update the application status as fast as possible and with ease.  Wants to always have up-to-date data about the application so accurate matches are made between applications and donors. | |
| **Pre-Conditions:** | The Application must exist within the database | |
| **Post-Conditions:** | The Application entry is updated into the APPLICATION table  Generated confirmation message confirming the updates made to the application  Confirmation message, pertaining the details made to the application, sent | |
| **Flow of Activities:** | 1. Applicant/Administrator requests to Update Application 2. Applicant/Administrator enters the application reference number 3. Applicant/Administrator makes the necessary changes to the Application | * 1. Prompts the Applicant/Administrator to enter the application reference number   2. Invokes Read Application   3. Displays the details of the application   4. Prompts the Applicant/Administrator to make changes to the displayed details   5. Updates the Application record in the database   6. Displays a message for the user that the update has been completed   7. Generates confirmation message   8. Sends confirmation message pertaining the updates made |

# 8.2 Use case diagram for Update Application



# 8.3 System sequence diagram for Update Application

This diagram is a representation of the communication between the system and the applicant/ administrator during the Update Application process.

