Milestone1

Bohlokoa tilo (1852529)

bolekwa mboniswa (1397266)

xolani mAGOLOZA (1412049)

CHICO VICENTE (1844374)

RAHUBE SEABI (1834316)

# Purpose

This document is the first iteration in mapping and showing our current understanding of your

organization’s domain. We will accomplish this by identifying what we believe is the main problem that needs to be solved; then we will demonstrate our understanding of the domain through a pair of data analysis diagrams. The diagrams will map our current understanding of your domain’s stakeholders and the different tasks that you need our proposed solution to execute.

# Executive Summary

We have identified the problem in your system to be the need for a system that optimally match donors and students

We used a Class Diagram to model the domain of the client, and discovered that the main classes of the database will be: Donor, Administrator, Applicant, Application and Donation

We then modelled the system usage through Use Case Set and Use Case Diagram based off the classes. The administrator will be granted access to most of the uses as they process the matches and deal with the system on behalf of the donor. The student will only have access to creating their applications and profiles.

We also defined the main features of the system. Amongst which is the system being able to aid and manage the process of matching donors and applicants. Another important feature is the system being able to generate overview reports on the system effectiveness; also processing the data and displaying it.

We explored different methods for requirement elicitation, with Requirement Workshops emerged as the clear favorite. This is because it gives access to a lot of stakeholders in a limited amount of time. We supplemented this with more user specific methods like interviews and questionnaires.

We broke down some of their use cases to give you a clear functional requirements. We also detailed 5 non-functional requirements that will make the system more reliable. We identified that a big constraint to us building an optimal system is that we have to build a desktop application specifically for the kiosks (as opposed to a multi-platform system).

# Problem Statement

We must provide a solution that allows #fundMe administrators to optimally match donors and students in need, then help them manage and monitor such relationships once formed.

**Class Diagram representation of the #fundme system’s database**

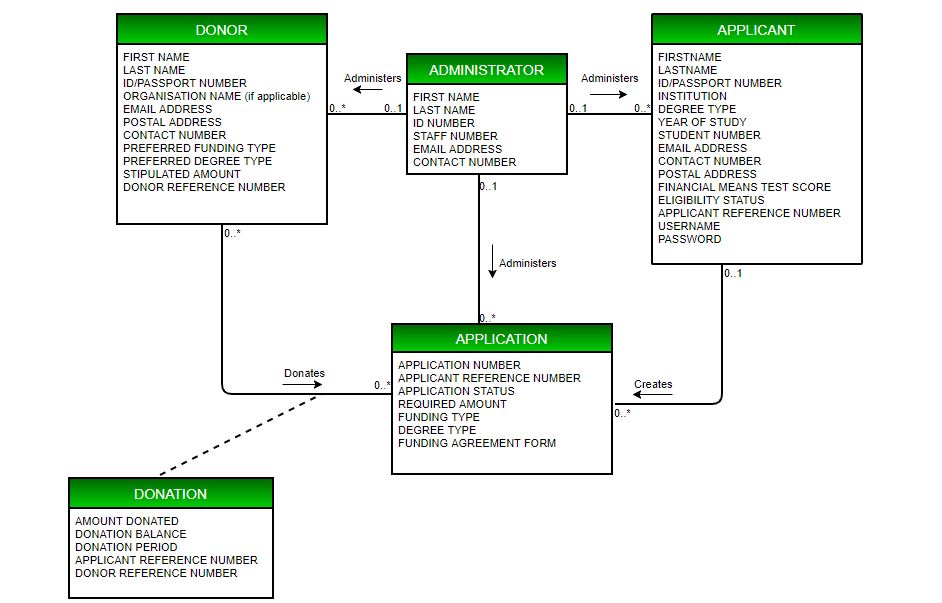
****

Figure 1: Class Diagram of the relationships within the system’s database

The purpose of the class diagram is to conceptualize the business rules and provide us with an overview of the relationships between the entities of the #fundme database system.

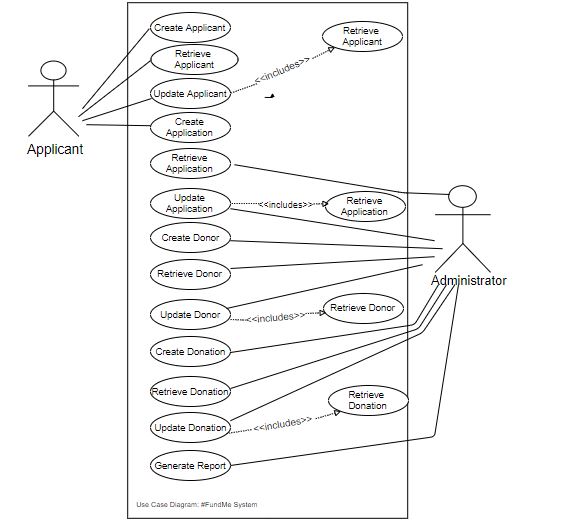
**CRUD Use Case Set of #fundme database system**

|  |  |  |  |
| --- | --- | --- | --- |
|  | Use Case | Description | Used use cases |
| APPLICANT | | | |
|  | Create Applicant | Creates new applicant |  |
| Retrieve Applicant | Retrieves applicant’s record |  |
| Update Applicant | Updates applicant’s demographic information | Retrieve applicant |
| APPLICATION | | | |
|  | Create Application | Creates new application |  |
| Retrieve Application | Retrieves application from database |  |
| Update Application | Updates application information | Retrieve application |
| DONOR | | | |
|  | Create Donor | Creates new donor |  |
| Retrieve Donor | Retrieves donor’s record |  |
| Update Donor | Updates donor’s information | Retrieve Donor |
| DONATION | | | |
|  | Create Donation | Creates new donation |  |
| Retrieve Donation | Retrieves donation information |  |
| Update Donation | Updates donation information | Retrieve Donation |
| ADMINISTATOR | | | |
|  | Create Administrator | Creates new administrator |  |
| Retrieve Administrator | Retrieves administrator’s information |  |
| Update Administrator | Updates administrator’s information | Retrieve Administrator |
| REPORT | | | |
|  | Generate report | | |

# **Use case diagram for #fundme database system**

Why Use Case Diagram? It is a simplified visual tool that demonstrates system functionality from the users’ perspective. This allows for their perspective on whether the business processes have been modelled properly, as it’s easy for them to understand our very technical approach to the solution.

Assumptions: We assumed we will need to include the “Retrieve” use cases for the “Update” use cases. Donor is absent from the diagram because they are not part of the system.



|  |  |
| --- | --- |
| Table of the system Features and Benefits | |
| Feature | Benefits |
| Aid and manage the process of matching donors and applicants | This will ensure that students find the right type of funding, and the donors fund the type of students they are most suitable for. This will speed up the matchmaking process significantly for administrators and increase the accuracy. |
| Application/Registration for applicants and registration for donors | This aids the collection of data on the applicants and donors. The information collected from this will help the matchmaking process. (e.g. filter students according to the type of funding they need) |
| Using advanced plug-ins for advanced data manipulation and information presentation | As we stated in Milestone 0, our teams’ approach is richness of data. We will be using that in this feature to help generate well presented graphs and info visualisation. |
| Generating Reports | This will help us get an overview of system and organization performance.  The information gained will aid in the decision making |
| Managing and keeping track of donor – student relationships | This will help gather information for generating reports  This will help track that both parties are keeping to the terms of the agreement |

# Requirement Elicitation

* **General:** Our team has chosen requirement workshops as the base elicitation technique as we deem it to be the most effective and suitable to a system with such varied stakeholders. With a requirement workshop that is well facilitated and documented; you are able to get a lot of information out of the stakeholders in the shortest possible space of time. It would be ideal to have administrators, as well as a couple of representatives from the donor and student side. This means that you can get multiple perspectives to the requirements, and the stakeholders interacting with each other can help us make more informed decisions on compromises (e.g. If the administrators want a feature that is not entirely to the liking of the students). We also aim to set up the workshops hours in a way that best suits the different schedules of the stakeholders.
* **Donors:** This stakeholder will interact with the system the least, as the administrator will mostly be interacting with the proposed system on their behalf. We will also have the least access to them as we can assume, they have less time to give to our elicitation process, and location constraints are also an evident obstacle. The best solution for them is to send all of them electronic surveys and questionnaires. This will be adequate for the limited input we need from them and a perfect work around the constraints.
* **Administrators and Applicants:** A collaboration of different requirements elicitation methods will prove useful in the case of the administrators and applicants, as no single technique will be adequate and effective enough to work for both stakeholders. These methods include the abovementioned requirement workshops as well as interviews for a more intimate setting. Structured individual interviews with a number of stakeholders will aid in attaining their explicit requirements and needs, some of which may possibly be of great assistance to the elicitation process. Observation is also part of our collaboration of requirements gathering strategies. Observation can either be passive or active, and we aim to use the active kind of observation. The active approach to observation involves a more hands on way to closely analyze the varied systematic needs and experiences of the stakeholders. This method will prove more useful once we have some form of prototype system in place or during the early stages of the software system release. Then will we be able to actively observe how the users interact with the system, and thereafter have them review the system. Therefore active observation allows us to put ourselves in the users’ shoes and to experience the system as they do and this will also help us, as analysts, to notice those requirements that may not be so obvious to either of the stakeholders. This kind of observation again works in our favor as it allows for us to directly ask the stakeholders questions about their experience as they use the system. Although the continuous questioning may prove to be quite perturbing to the subject of our observation, it is very necessary for us to ask the “whys” and “hows” since these questions are most analysts’ bread-and-butter and not mention the most vital way to gather requirements in this situation. However, in the case of the administrator we will have to limit our questions to avoid taking up too much their productivity time.

Sample Requirements Definition

# Functional Requirements

**Create Application:**

* System must allow the student to create the application
* System must allow applicant to add details on the type of application
* System must update the application to the database

**Create Donation:**

* System must match an applicant with a donor
* System must store details of the donation to the database

**Retrieve Donor:**

* System must allow the administrator to access donor information on the database
* System must allow filtering when searching for a donor

# Non-Functional Requirements

* System must create a match between donor and applicant within 5 days of a submitted application’
* System must verify the student details with university databases
* System must elicit information from the students about the donor’s efficiency on a quarterly basis
* System must lock out users when they’re inactive for more than 5 minutes
* System must request the student to rate and review the system after a successful application

# Constraints

* System must only be made for use at the client’s kiosks
* System must make use of a desktop application only (when accessibility can be increased by accommodating for mobile apps)