**Mentor Management System**

**Requirements Analysis**

**1. INTRODUCTION**

**PURPOSE OF THE SYSTEM**

The transition between High School and University can be such a grueling experience. Students encounter many challenges during the year; many of them are unable to overcome those challenges because of a lack of proper guidance. There exists mentorship programs on campus to tackle those problems, but most of them have proved ineffective. The main reasons are because prospective mentees are not motivated enough to walk to a particular location in order to sign up for the program and because of an inefficient follow up structure. The Mentor Management application is being developed to bridge that gap. It aims to create an efficient system that allows mentees and mentors to sign up at their comfortable location and using any device. It also enables mentees to receive briefings and schedule meetings from the application. This application serves as a connection between mentees and mentors.

**SCOPE OF THE SYSTEM**

The system allows screen size responsiveness and can thus run on any kind of device including desktop computers and mobile devices. It is designed as a self-service application where mentees and mentors can fill information themselves. It also comprises an administrator whose job is limited to monitor the transactions between mentors and mentees. A help menu is incorporated for users who are struggling to use the application. In developing the application we assume that prospective mentees know the importance of getting mentored, the job of making students aware is generally carried out by the Counselling and Career Department of the university. The system is built only for the purpose of facilitating fast and reliable sign up process. Although the application enables scheduling of meetings, those meetings are to be held physically, that is, the pair has to meet in person. The system does not allow online meeting through audio or videoconferencing.

**OBJECTIVES AND SUCCESS CRITERIA OF THE PROJECT**

The system will be considered a success if:

* It allows any kind of users (Administrator, Mentor, and Mentee) to sign up and login.
* It successfully captures the users’ information into the database.
* It facilitates the matching of Mentor-mentee according their background.
* It is able to schedule meetings of the pair.

**DEFINITION, ACRONYMS AND ABBREVIATION**

Application System: An application program (app or application for short) is a computer program designed to perform a group of coordinated functions, tasks, or activities for the benefit of the user.

System administrator: A person who is responsible for managing a multi-user computing environment.

Mentor: A person who gives a younger or less experienced person help and advice over a period of time, especially at work or school.

Mentee: Someone who is given support and advice about their job by a mentor. Screen size responsiveness: A design method that allows tailoring of content to different screen sizes. User-friendly System: A system that is easily operated and understood by means of a straightforward guide in jargon-free language.

**REFERENCES** The above definitions were taken from the CAMBRIDGE ONLINE DICTIONARY and Google-enabled dictionary.

**OVERVIEW**

This Requirement Analysis Document provides the description of the general outlook of the Mentor Management Application. The proposed solution of the system and all the functional and non-functional requirements are found below to help the stakeholders have a full understanding of the system which will deliver SSS Mentor-mentee services.

**2. Current System**

Currently there is no system that can help university students find a mentor. A mentor is necessary to assist students know what careers correspond to the degree that they are studying or hope to study, what combination of courses are recommended for a career their hoping to pursue and overall give the particular student advice on their professional life. Because tertiary is very challenging and confusing for a lot of students, guidance from a more experienced individual that has either been through the same challenges and was able to rise above them or someone who has achieved this student’s set goals, would prove very resourceful to a majority of students. For instance, a student studying a BSc in Biology might have passion about nature and animals but in the midst of their university career start to wonder what they could do with their qualification, or whether they need to study further to enter their career of choice. A lot of students spend most of their time on their mobile smart phones/tablets and mentees spend more of their time in front of a computer which is why the system is a web application.

**3.1. Functional Requirements**

**Mentors**

* Need accurate information about mentee; to accept a mentee they want to help.
* Needs to be able to opt-out of mentoring more mentees; if they can’t help anymore mentees.
* Needs to be able to deactivate their system account.
* Needs to see when their mentorship is requested; to know whose profile to view.
* Needs to be able to accept request; so mentee knows to contact them.
* Needs a user-friendly system.
* Efficiency
* Quick Responsiveness.

**Mentee**

* Needs accurate information about mentor; to make a better informed choice.
* Needs to find a mentor that matches their preferences; to get an effective mentor.
* Needs to know that the mentor is credible; for safety reasons and effectiveness.
* Needs to indicate interest in being mentored by particular mentor.
* Needs feedback on whether mentor is interested; to contact the mentor.
* Needs to be able to deactivate their system account.
* Needs a user-friendly system.
* Constant interaction and communication with their mentor.
* Efficiency.
* Quick responsiveness.
* Needs accurate storage of their personal data; mentor is matched more accurately.

**Administrator**

* View feedback about mentor; to know which mentors need help or whose accounts to deactivate.
* Notifications when a new mentor registers to system; so they can verify them.
* View mentor profiles; to use to verify the mentor.
* Needs to be able to deactivate users that are misusing the system
* Have to verify the mentor on the system; mentees need verified mentors.

|  |  |  |
| --- | --- | --- |
| USER | FUNCTIONAL REQUIREMENTS | PRIORITY |
| General | System must allow user to register and specify the type of user (Mentor/Mentee). | H |
| System must allow user to login using a username and password | H |
| System must allow user to deactivate their account | M |
| System must allow user to update their profile. | M |
| Mentee | System must allow mentee to fill out a form detailing preferred mentor. | H |
| System must allow mentee to update their mentor preferences. | M |
| System must allow mentee to search for a mentee; provide filtered suggestion based on preferences. | H |
| System must allow mentee to view a mentor’s profile (including rating). | H |
| System must allow mentee to request a mentor’s mentorship. | H |
| System must allow mentee rate a mentor; provided the two are paired therefore if mentor has accepted the request. | H |
| System must allow mentee to mentor’s contact details; provided mentor has accepted the request. | H |
| Mentor | System must allow mentor to fill out a form detailing preferred mentee. | H |
| System must allow mentor to receive request for mentorship. | H |
| System must allow mentor to view a mentee’s profile | H |
| System must allow mentor to accept a request; pairing the two. | H |
| Administrator | System must allow admin to verify new mentors. | H |
| System must allow admin to view mentor ratings. | M |
| System must allow admin to deactivate a mentor’s or mentee’s account. | M |

**3.2. Non-functional Requirements**

**Usability**

* All delays in the system longer than 0.5 seconds will produce a dialog box that says "Please wait."
* It is possible to reach any given system function from the main window in less than 4 clicks.
* It is possible to accomplish any given task with just the keyboard, without the mouse.
* All buttons in the system will adhere to established button convention (link to established button convention regarding size, naming, position, etc.)
* All screens will have a help button. Each help button on a given screen must provide at least one 'topic' for each control on the screen.

**Reliability**

* The system will check the all operator inputs to see that they fall within their required ranges.
* The system will check all disks for bad blocks each time it is booted.

**3.3. System Models**

**3.3.1 Scenarios**

* Register User
* Login User
* Create profile
* Create pair
* Rate Mentor

**3.3.2 Use case model**

**Create pair**

|  |  |  |
| --- | --- | --- |
| Use case name:  UC 1 | Create Pair | |
| Scope: | Mentorship Program | |
| Triggering event: | Mentee and mentor profiles uploaded onto the system | |
| Brief description: | The mentee selects a mentor from a list of filtered mentors based on their preferences. The mentee can then view the mentors profile and make a decision on who to select. Once a mentor is selected, the mentee and mentor get allocated to a pair. The pair status is set to active, pair start date is updated. | |
| Actor(s): | Mentee | |
| Related use cases: | View Mentor  Update mentee | |
| Stakeholders & interests: | Mentee: Reliable and intuitive way of creating a pair  Mentor: Being allocated to the pair and receive a mentee request. | |
| Pre-conditions: | Mentees must be created  Mentees must have updated profile and chose their preferences  Mentors must have updated their profile | |
| Post-conditions: | Pair number generated and stored in the Pair datastore  Mentee and mentor stored in Pair datastore  Pair Status set to active  Increase number of mentees in Mentor datastore by 1  Update pair start date | |
| Flow of activities: | Actor | System |
| 1. Mentee wants to select mentor  2. Selects mentor | 1.1.) Display filtered list of mentors based on at least one preference  2.1) Invoke View Mentor use case  2.2) Confirmation of use case invoked  2.3) Pair number generated and stored in the Pair datastore  2.4)Mentee and mentor stored in Pair datastore  2.5)Pair Status set to active  2.6) Update pair start date  2.7) Increase number of mentees in Mentor datastore by 1  2.8) Generate and send email to mentor and mentee with their pair number  2.9) Display confirmation of mentor selected. |
| Extensions |  | * 1. If mentee had not updated profile (selected preferences)      1. Invoke Update mentee use case      2. Confirmation of use case invoked |

**Register User**

The same process happens for Mentee. The use case will assume Mentor being registered.

|  |  |  |
| --- | --- | --- |
| Use case name:  UC 2 | Register User | |
| Scope: | Mentorship Program | |
| Triggering event: | Mentor request to register | |
| Brief description: | The Mentor request to be registered and be able to be paired with a mentee based on the Job industry specified upon registration. | |
| Actor(s): | Mentor | |
|  |  | |
| Stakeholders & interests: | Mentor: Reliable and intuitive way of creating a profile upon registration  Mentee: Being able to find registered mentor | |
| Pre-conditions: | Mentor must be a specialist in a specific field to be able to complete registration. | |
| Post-conditions: | Mentor profile updated on mentor data store with province and area of specialization included. | |
| Flow of activities: | Actor | System |
| 1. Mentor request to register  2. Fill out the form and submit | 1.1.) Display registration form to fill out.  2.1) Save information to the database  2.2) Confirmation of successfully submission |
| Extensions | 2a. Fill out the requested fields on the form and submit | 2a.1) If mentor did not fill out the Job industry and province field the system does not confirm submission instead mentor is requested to fill out those fields as required.  2a.2) Confirmation of successfully submission |

**Rate Mentor**

|  |  |  |
| --- | --- | --- |
| Use case name:  UC 3 | Rate Mentor | |
| Scope: | Mentorship Program | |
| Triggering event: | End of the year evaluation | |
| Brief description: | The mentee request to rate mentor at the end of their interaction throughout the year. | |
| Actor(s): | Mentee | |
| Related use cases: | Create Pair | |
| Stakeholders & interests: | Administrator: To access the effectiveness of mentorship program  Mentee: To express their opinion about their mentor | |
| Pre-conditions: | Mentee and Mentor must have been paired throughout the year. | |
| Post-conditions: | The rating for that year be stored on the database | |
| Flow of activities: | Actor: Mentee | System |
| 1. Selects mentor rating  2. Selects rating value and submit | 1.1.) Display rating on an integer scale.  2.1) Value stored in the database  2.2) Confirm successful submission. |

**Use Case Diagram**

As per functional requirements, this diagram depicts which use cases will be used by which user. These can be interpreted as methods and the users that those methods are implemented for.



**Main Functions Sequence Diagrams**

***Update Mentor Verification***

If a mentor has registered onto the system, they do not appear on mentee’s search list at all unless it has been verified somehow by the administrator that they are who they say they are on the system. A post condition for this use case is that the verified Boolean attribute for the mentor entity will be set to true, if the mentor is indeed credible.



***Create Preference List***A precondition for creating a preference list is that the mentee must be registered on the system thus in the Mentee data store. And a post condition is that their preference list must be stored in the PreferenceList data store thereafter.



***Search Mentor***



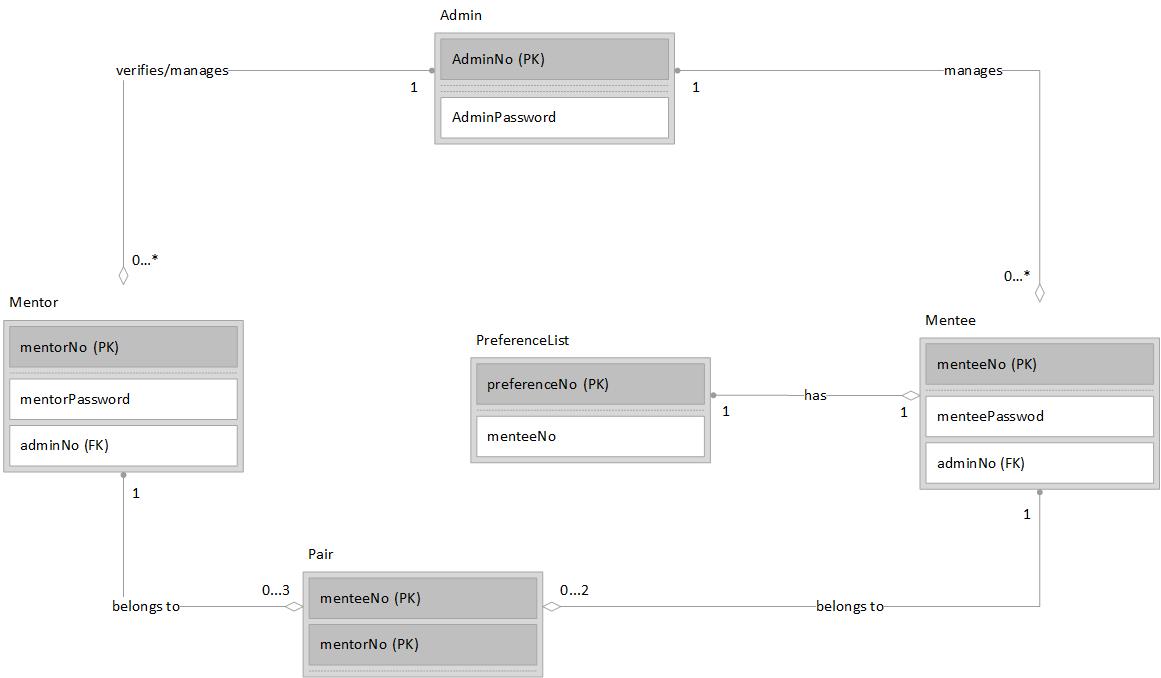
A mentee’s search list consists of a list of mentors whose profile matches the preferences of that mentee by at least 4 characteristics. This list is ordered by the number of preferences matched with the highest number on top. A precondition for all listed mentors is that they have to be verified.

***Accept Request***

A precondition for this use case is that a request has to be sent first by the mentee. A post condition is that this mentor and mentee will now be paired thus included in the paired entity, allowing the mentee to be able to rate the mentor.

**3.3.3 Object Model**

The model below depicts the object relationship identified in the system.



**3.3.4 Dynamic model**

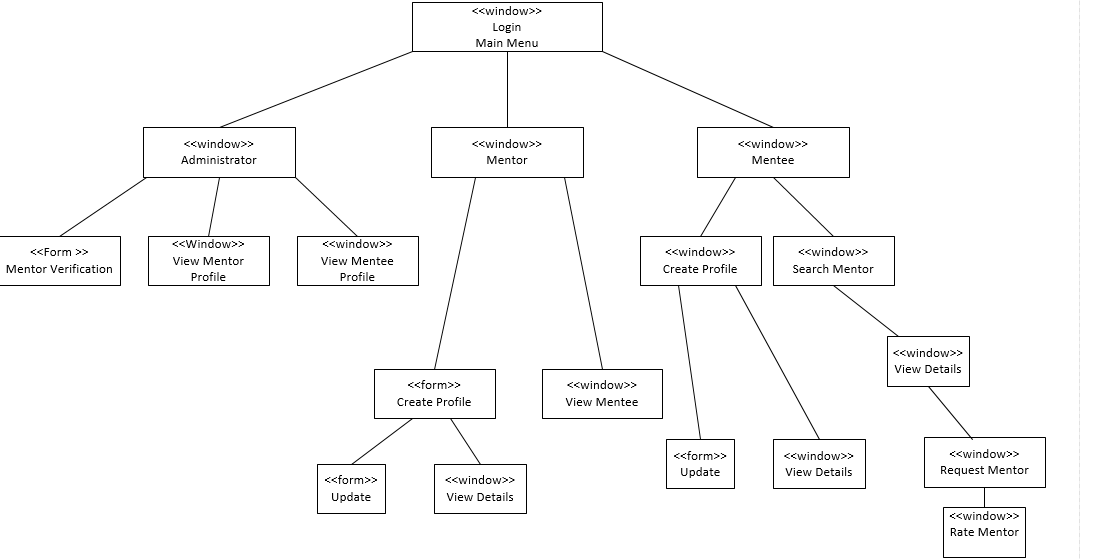
**Activity Diagram**

An activity diagram is a graphical representation of the flow of control between the different actors in the Mentorship program. Activity diagrams also show the data that is being passed around and stored by the Mentorship program. The sequence in the activity diagram is derived from the flow of activities section of the fully dressed use case description therefore each activity diagram focuses on a specific use case.

**Create pair activity diagram**

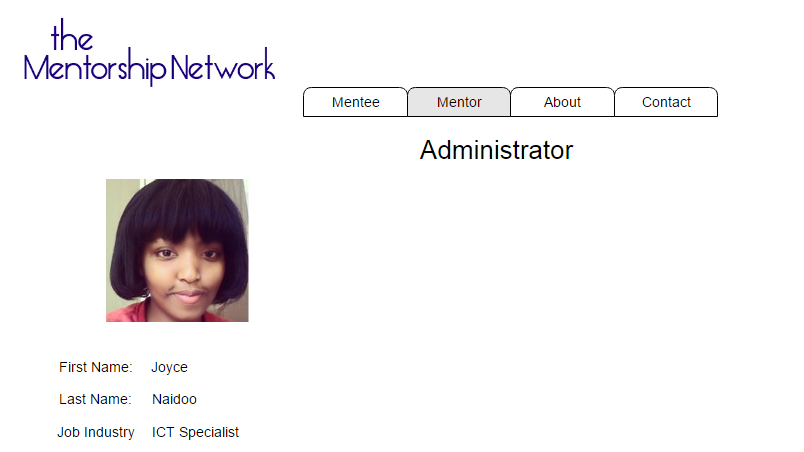
**3.3.5. User interface—navigational paths and screen mock-ups**

**Navigational Paths**

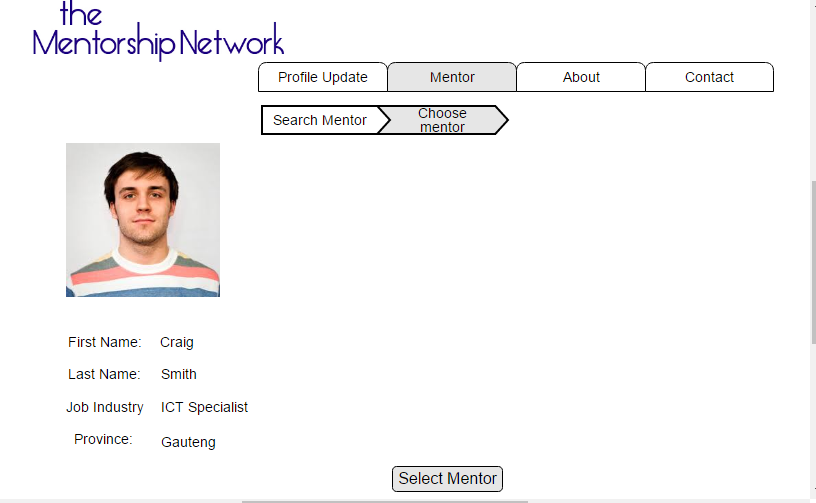
****

**Screen Mock-up**

Below is the screen mock up for the Administrator under mentor tab.



**Mentee Screen Mock up**

****

**4. GLOSSARY OF TERMS**

|  |  |
| --- | --- |
| Term | Meaning |
| UC (1) | Use Case number priority |
| Mentor | User of the system to be paired |
| Mentee | User of the system to be paired with Mentor |
| Data store | Denotes Database |
| Object Model | Depicts class diagram |
| Priority H | High Priority |
| Priority M | Medium Priority |