

ULTRON III

Elaboration Document

*FORAGE*

Team Members:

|  |
| --- |
| 216198607, Zandile Dube  217055443, Lavhelesani Mamphwe  216673380, Emandleni Moyo |

Document Due Date: 20 JUNE 2019

**TABLE OF CONTENTS**

[1 FUNCTIONAL REQUIREMENTS 3](#_Toc14564041)

[1.1 Analysis Use Case Model 3](#_Toc14564042)

[1.2 Use Case Glossary and Responsibilities 4](#_Toc14564043)

[2 UI Prototypes 6](#_Toc14564044)

[2.1 Team UI Guidelines 6](#_Toc14564045)

[2.2 UI Designs & Updated Analysis Use Case Narratives 7](#_Toc14564046)

[2.2.1 Designed by 7](#_Toc14564047)

[2.2.2 Designed by 10](#_Toc14564048)

[2.2.3 Designed by 12](#_Toc14564049)

[2.2.4 Designed by 13](#_Toc14564050)

[2.2.5 Designed by 15](#_Toc14564051)

[2.2.6 Designed by 17](#_Toc14564052)

[2.2.7 Designed by 19](#_Toc14564053)

[2.2.8 Designed by Zandile Dube 20](#_Toc14564054)

[2.2.9 Designed by Zandile Dube 21](#_Toc14564055)

[2.2.10 Designed by Emandleni Moyo 22](#_Toc14564056)

[2.2.11 Designed by Emandleni Moyo 24](#_Toc14564057)

[2.2.12 Designed by Emandleni Moyo 26](#_Toc14564058)

[2.2.13 Designed by Emandleni Moyo 28](#_Toc14564059)

[2.2.14 Designed by Zandile Dube 29](#_Toc14564060)

[2.2.15 Designed by Lavhelesani Mamphwe 30](#_Toc14564061)

[3 DATA REQUIREMENTS 31](#_Toc14564062)

[3.1 Domain Class Diagram 31](#_Toc14564063)

[3.2 Implementation Ready Class Diagram 33](#_Toc14564064)

# FUNCTIONAL REQUIREMENTS

## Analysis Use Case Model

## Use Case Glossary and Responsibilities

|  |  |
| --- | --- |
| **Team Member Responsible:** | |
| **Use Case Id** | **Use Case Name** |
| A0100 | **Register** |
| A0200 | Login |
| A0300 | Logout |
| A0400 | Deregister |
|  |  |
|  |  |
| **Queries/Reports** | |
|  |  |

|  |  |
| --- | --- |
| **Team Member Responsible:** | |
| **Use Case Id** | **Use Case Name** |
| B0100 | View Labs |
| B0200 | Find Lab |
| B0300 | **View Tutors List** |
| B0400 | **View Tutor Profile [Student View]** |
| B0500 | **Request Tutor** |
| B0600 | Manage Requests |
| **Queries/Reports** | |
|  |  |

|  |  |
| --- | --- |
| **Team Member Responsible:** | |
| **Use Case Id** | **Use Case Name** |
| C0100 | View Profile |
| C0200 | Manage Profile |
| C0300 | View Requests |
| C0400 | Manage Requests |
| **Queries/Reports** | |
|  |  |

|  |  |
| --- | --- |
| **Team Member Responsible:** | |
| **Use Case Id** | **Use Case Name** |
| D0100 | Manage Computer Labs |
| D0200 | Manage Tutors |
| D0300 | Mange Requests |
| D0400 | Manage Machines |
| **Queries/Reports** | |
|  | Determine vacant machines in a designated computer lab. |
|  | Determine the most requested tutor, number of requests made to that specific tutor and their outcomes. |
|  |  |

# UI Prototypes

## Team UI Guidelines

## UI Designs & Updated Analysis Use Case Narratives

### Designed by

|  |  |  |
| --- | --- | --- |
| **Use Case ID** | **Use Case Name** | |
| **A0100** | **Register** | |
| **Primary Business Actors** | | **Other participating Actors** |
| Students, Tutors | |  |
| **Description** | First time users must register onto the information system. They must specify their role preference before registration. Upon a successful registration, the user will then be able to log into the system and will only be allowed to use the allocated features dedicated to their role. | |
| **Pre-Conditions** | To register for any role, the user must not have registered before. A user can only employ one username identity. | |
| **Triggers** | On click of the register button. | |
| **Post-Conditions** | User is registered onto the information system and redirected to a default role specific form. The user will have to wait for approval from the Administrator, for the modules selected on the “Add Modules” form before they can receive requests. | |
| **Basic Flow of Events** | 1. User launches application. 2. User is directed to the login form. 3. User leaves input fields empty. 4. User selects the “Student” role from the dropdown box. 5. User clicks on register. 6. User is directed to the student role specific input form. 7. User provides appropriate credentials. 8. User clicks on register. 9. A message confirms success. 10. User is redirected to a role specific default form. | |
| **Alternate Flow of Events** | 1. User launches application. 2. User is directed to the login form. 3. User leaves input fields empty. 4. User selects the “Tutor” role from the dropdown box. 5. User clicks on register. 6. User is redirected to the tutor role specific input form. 7. User provides appropriate credentials. 8. User clicks “Next”. 9. User is directed to a “Add Modules” form where modules and checkboxes are displayed. 10. User selects the modules they wish to tutor. 11. User clicks on register. 12. A message confirms success. 13. User is redirected to a role specific default form. | |
| **Alternate Flow of Events** | 6. User provides incorrect credentials.  7. User clicks on submit.  8. A message signals failure.  9. User is prompted to re-enter the form with correct details.  10. Basic flow of events number 6. | |
| **Initial UI design** |  | |

### Designed by

|  |  |  |
| --- | --- | --- |
| **Use Case ID** | **Use Case Name** | |
| A0200 | Login | |
| **Primary Business Actors** | | **Other participating Actors** |
| Students, Tutors and Administrators | |  |
| **Description** | Registered users are able to log into the system. Each registered user must use their own credentials in order to login. | |
| **Pre-Conditions** | To login, the user must be registered i.e. the information system must be aware that such a user exists. | |
| **Triggers** | On click of the login button. | |
| **Post-Conditions** | The user is logged into the information system and redirected to a default role specific form. | |
| **Basic Flow of Events** | 1. User launches the application. 2. User is redirected to the login form. 3. User provides appropriate credentials. 4. User clicks on login. 5. A message confirms success. 6. User is redirected to a role specific default form. | |
| **Alternate Flow of events** | 1. User provides incorrect credentials. 2. User clicks on login. 3. A message signals failure to login and prompts re-entry. 4. Basic flow of events number 2. | |
| **Initial UI design** | C:\Users\Admin\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Login Form.png | |

### Designed by

|  |  |  |
| --- | --- | --- |
| **Use Case ID** | **Use Case Name** | |
| B0100 | View Labs | |
| **Primary Business Actors** | | **Other participating Actors** |
| Students and Tutors | |  |
| **Description** | Students and tutors are able to view computer labs. | |
| **Pre-Conditions** | To view labs, the user must be logged onto the system. | |
| **Triggers** | On click of the ‘view labs’ button. | |
| **Post-Conditions** | A floor layout plan of all computer labs is displayed along with other relevant data. | |
| **Basic Flow of Events** | 1. User clicks on ‘View Labs. 2. Data about computer labs is displayed on screen. | |
| **Initial UI design** |  | |

### Designed by

|  |  |  |
| --- | --- | --- |
| **Use Case ID** | **Use Case Name** | |
| B0200 | Find a Machine | |
| **Primary Business Actors** | | **Other participating Actors** |
| Students and Tutors | |  |
| **Description** | Students and tutors must be able to navigate to any given machine within any computer lab. | |
| **Pre-Conditions** | To find a machine, the user must be on the View Labs form and choose to search for a machine. | |
| **Triggers** | User searches for a machine by providing a unique machine identifier. | |
| **Post-Conditions** | The computer labs floor layout plan is narrowed down to only display the specific lab in which the machine searched for is found and other relevant data. | |
| **Basic Flow of Events** | 1. View Labs form is shown. 2. User provides a unique machine identifier. 3. User is redirected to the specific lab in which the machine searched for is found, but mainly giving attention to the searched machine’s data. | |
| **Alternate Flow of events** | 1. User provides an invalid machine identifier. 2. The system signals an invalid input notification and prompts re-entry 3. Basic flow of events number 2. | |
| **Initial UI design** |  | |

### Designed by

|  |  |  |
| --- | --- | --- |
| **Use Case ID** | **Use Case Name** | |
| **B0300** | **View Tutors List** | |
| **Primary Business Actors** | | **Other participating Actors** |
| Students | |  |
| **Description** | Users must be able to view approved tutors for each module. | |
| **Pre-Conditions** | To View Tutors, the user must be logged in. There must be at least one tutor for a module to be displayed. | |
| **Triggers** | User clicks on ‘view tutors’ button. | |
| **Post-Conditions** | A list of tutors for each module is shown to the user. | |
| **Basic Flow of Events** | 1. User clicks on the view tutors button. 2. Student selects the module for which the tutorials are required. 3. A list of available tutors appears on the screen for the matching module selected. | |
| **Initial UI design** |  | |

### Designed by

|  |  |  |
| --- | --- | --- |
| **Use Case ID** | **Use Case Name** | |
| **B0400** | **View Tutor Profile [Student View]** | |
| **Primary Business Actors** | | **Other participating Actors** |
| Students | |  |
| **Description** | Users must be able to view a chosen tutor from the list of tutors. | |
| **Pre-Conditions** | Users must be signed in. There must be at least one tutor for a module to be displayed.  The user must choose to view a tutor. | |
| **Triggers** | Upon double clicking on the tutor record. | |
| **Post-Conditions** | The chosen tutor’s profile will be displayed from which further action may be taken. | |
| **Basic Flow of Events** | 1. User is on the ‘View Tutors’ form. 2. User double clicks on the preferred tutor. 3. The chosen tutor’s profile form appears. | |
| **Initial UI design** |  | |

|  |  |  |
| --- | --- | --- |
| **Use Case ID** | **Use Case Name** | |
| **B0500** | **Request Tutor** | |
| **Primary Business Actors** | | **Other participating Actors** |
| Students | |  |
| **Description** | Users must be able to request a tutor for a tutorial. | |
| **Pre-Conditions** | Users must be signed in. There must be at least one tutor for a module to be displayed.  The user must choose to view a tutor. The tutor profile must be displayed. | |
| **Triggers** | Upon clicking the request button. | |
| **Post-Conditions** | A request will be sent to the chosen tutor and the user will see a dialog box asking them to confirm the request. | |
| **Basic Flow of Events** | 1. User is viewing the chosen tutor’s profile. 2. User clicks on the request button. 3. A dialog box will be displayed asking the user to confirm the request. 4. The user will click the send button. 5. Another dialog box will be displayed stating that the request has been sent. 6. The user clicks ok. 7. The user is redirected to the tutor list. | |
| **Initial UI design** | 1. The user clicks cancel. 2. Basic Flow of Events number 7. | |

### Designed by

|  |  |  |
| --- | --- | --- |
| **Use Case ID** | **Use Case Name** | |
| C0100 | Accept/ Decline Requests | |
| **Primary Business Actors** | | **Other participating Actors** |
| Tutor | |  |
| **Description** | A tutor can either accept or decline a tutorial request from a student. | |
| **Pre-Conditions** | Tutor must be registered  Tutor must be logged onto their account  Tutor must have received a request from a student for a tutorial | |
| **Triggers** | The tutor has to accept or decline a request from a student. | |
| **Post-Conditions** | Tutor request outcome changes and the student is able to rate the tutor immediately | |
| **Basic Flow of Events** | 1. A0200 is invoked and the tutor lands on the tutorial requests screen 2. Tutor selects the module for which requests are made. 3. Student request is either accepted/ declined for the tutorial 4. The request status changes based on the decision made | |
| **Alternate Flow of events** | In case user does not have requests, they will be presented with no data. | |
| **Initial UI design** |  | |

### Designed by Zandile Dube

|  |  |  |
| --- | --- | --- |
| **Use Case ID** | **Use Case Name** | |
| C0200 | Edit Profile | |
| **Primary Business Actors** | | **Other participating Actors** |
| Tutor | |  |
| **Description** | A tutor can either update or review information on their profile. | |
| **Pre-Conditions** | Tutor must be registered  Tutor must be logged onto their account | |
| **Triggers** | The tutor will choose to update or review the information on their profile. | |
| **Post-Conditions** | If changes are made to the tutor profile, fields are updated. The tutor can continue to review the profile, until they navigate to a different part of the system. | |
| **Basic Flow of Events** | 1. A0200 is invoked 2. The tutor lands on their profile screen 3. Navigate to Requests tab 4. Tutor selects the module for which requests are made 5. Student request is either accepted/ declined for the tutorial 6. The request status changes based on the decision made | |
| **Initial UI design** |  | |

### Designed by Zandile Dube

|  |  |  |
| --- | --- | --- |
| **Use Case ID** | **Use Case Name** | |
| C0300 | View Feedback | |
| **Primary Business Actors** | | **Other participating Actors** |
| Tutor | |  |
| **Description** | A tutor is able to see the feedback that their provided. | |
| **Pre-Conditions** | Tutor must be registered  Tutor must be logged onto their account  The tutor must have accepted a request from the student from which the feedback is from | |
| **Triggers** | The tutor must select on the tab with feedback | |
| **Post-Conditions** | The tutor is able to view their feedback | |
| **Basic Flow of Events** | 1. A0200 is invoked 2. The tutor lands on their profile screen 3. Navigate to Feedback tab | |
| **Alternate Flow of events** | In case user does not have feedback, they will be presented with no data. | |
| **Initial UI design** |  | |

### Designed by Emandleni Moyo

|  |  |  |
| --- | --- | --- |
| **Use Case ID** | **Use Case Name** | |
| D0100 | Manage Computer Labs | |
| **Primary Business Actors** | | **Other participating Actors** |
| Admin | |  |
| **Description** | Administrators have the ability to maintain computer labs within the system. Administrators have the authority to create, view, alter and/or delete computer labs, schedule, machines and the table layout plan of a computer lab. | |
| **Pre-Conditions** | The user must be registered on the system.  The user must be logged onto the system.  The user must be of type admin. | |
| **Triggers** | A new computer lab may need to be added or deleted  A computer lab’s information may need to be updated | |
| **Post-Conditions** | A computer lab has been added, updated or deleted  A computer labs’ schedule has been added, updated or deleted. | |
| **Basic Flow of Events** | 1. The administrator home screen is displayed  2. The user clicks on “Computer Labs”  3. A list of labs appear together with options to create, update or delete  4. The user must enter new data or manipulate existing data to enable the ‘Save’ button  5. The user must ‘Save’ to commit changes onto the system | |
| **Initial UI design** |  | |

### Designed by Emandleni Moyo

|  |  |  |
| --- | --- | --- |
| **Use Case ID** | **Use Case Name** | |
| D0200 | Manage Tutors | |
| **Primary Business Actors** | | **Other participating Actors** |
| Admin | |  |
| **Description** | Administrators have the ability to maintain tutor information. Administrators have the authority to create, view, alter and/or delete tutor information. | |
| **Pre-Conditions** | The user must be registered on the system  The user must be logged onto the system  The user must be of type admin | |
| **Triggers** | Tutor information may need to be verified, altered and/ or deleted | |
| **Post-Conditions** | A tutor has been added, updated or deleted | |
| **Basic Flow of Events** | 1. The administrator home screen is displayed  2. The user clicks on “Tutors”  3. A list of tutors appear together with options to manipulate the data  4. The user must enter new data or manipulate existing data to enable the ‘Save’ button  5. The user must ‘Save’ to commit changes onto the system | |
| **Initial UI design** |  | |

### Designed by Emandleni Moyo

|  |  |  |
| --- | --- | --- |
| **Use Case ID** | **Use Case Name** | |
| D0300 | Manage Students | |
| **Primary Business Actors** | | **Other participating Actors** |
| Admin | |  |
| **Description** | Administrators have the ability to maintain student information. Administrators have the authority to create, view, alter and/or delete students. | |
| **Pre-Conditions** | The user must be registered on the system  The user must be logged onto the system  The user must be of type admin | |
| **Triggers** | Student information may need to be verified, altered and/ or deleted | |
| **Post-Conditions** | A student has been added, updated or deleted | |
| **Basic Flow of Events** | 1. The administrator home screen is displayed  2. The user clicks on “Students”  3. A list of students appear together with options to manipulate the data  4. The user must enter new data or manipulate existing data to enable the ‘Save’ button  5. The user must ‘Save’ to commit changes onto the system | |

|  |  |
| --- | --- |
| **Initial UI design** |  |

### Designed by Emandleni Moyo

|  |  |  |
| --- | --- | --- |
| **Use Case ID** | **Use Case Name** | |
| D0400 | Report 1: Determine the number of vacant machines within a specific lab | |
| **Primary Business Actors** | | **Other participating Actors** |
| Admin | |  |
| **Description** | This use case creates a report for a list of vacant machines within a specific lab | |
| **Pre-Conditions** | Admin must be logged onto the system. | |
| **Triggers** | User wanted to find out which machines are currently vacant | |
| **Post-Conditions** | A report containing the list of vacant machine numbers in a particular lab | |
| **Basic Flow of Events** | 1. Admin enters query by selecting the computer lab from a drop-down box  2. System produces report. | |
| **Initial UI design** |  | |

### Designed by Zandile Dube

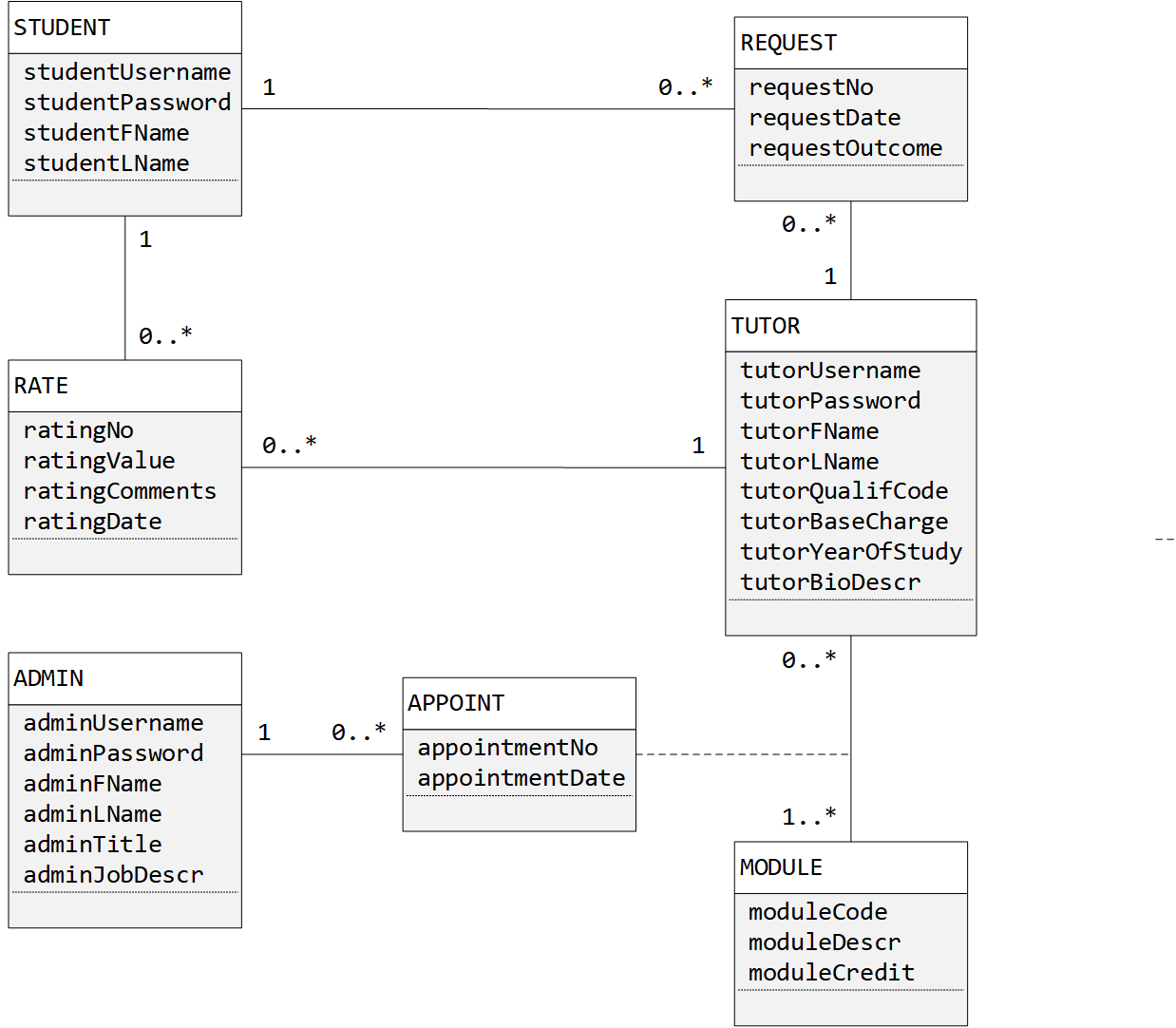
|  |  |  |
| --- | --- | --- |
| **Use Case ID** | **Use Case Name** | |
| D0500 | Report 2: Determine who is the most requested tutor | |
| **Primary Business Actors** | | **Other participating Actors** |
| Admin | |  |
| **Description** | This use case creates a report for a list of tutors sorted according to their number of requests for a specific module. | |
| **Pre-Conditions** | Admin must be logged onto the system | |
| **Triggers** | User wants to determine the most requested tutors according to a module | |
| **Post-Conditions** | A report containing the highest requested tutor for a module will be emailed to students. | |
| **Basic Flow of Events** | 1. Admin enters query by selecting the module from a drop-down box.  2. System produces report. | |
| **Initial UI design** |  | |

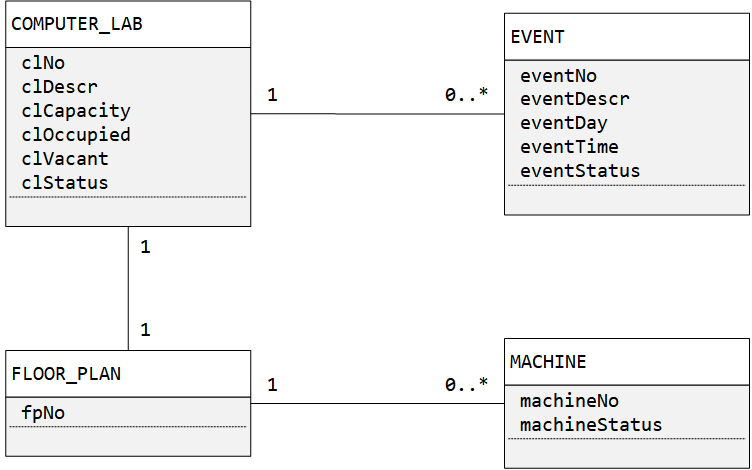
### Designed by Lavhelesani Mamphwe

|  |  |  |
| --- | --- | --- |
| **Use Case ID** | **Use Case Name** | |
| D0600 | Report 3: Determine the number of requests made for a module on the system | |
| **Primary Business Actors** | | **Other participating Actors** |
| Admin | |  |
| **Description** | This use case creates a report for a list of requests made for a specific module | |
| **Pre-Conditions** | Admin must be logged onto the system | |
| **Triggers** | User wants to determine the number of requests made for a module on the system | |
| **Post-Conditions** | A report containing the list of requests made for a module in descending order of requests | |
| **Basic Flow of Events** | 1. Admin enters query by clicking  2. System produces report | |
| **Initial UI design** |  | |

# DATA REQUIREMENTS

## 3.1 Domain Class Diagram





## Implementation Ready Class Diagram

