

CSY2030

Northampton Metropolitan University

Ana Rosa Gaspar 18400473 | BSc Computing |

Demo vídeo: <https://northampton.mediaspace.kaltura.com/media/Screen+Capture+CSY2030-+2019+May+05+04A12A40/1_okpy7jxp>

Table of Contents

[Introduction & Problem Specification 2](#_Toc7961348)

[Analysis 2](#_Toc7961349)

[Design 2](#_Toc7961350)

[Class Diagram 2](#_Toc7961351)

[Use Case Diagram 3](#_Toc7961352)

[Overall design 8](#_Toc7961353)

[System Description (Main components and functionalities) 8](#_Toc7961354)

[User guide 9](#_Toc7961355)

[Testing 13](#_Toc7961356)

[Conclusion 15](#_Toc7961357)

[References 16](#_Toc7961358)

# Introduction & Problem Specification

The Northampton Metropolitan University asked for a graphic user interface software system solution to enable their human resources department to keep track of lectures who work in various departments. The system should allow the secretaries of departments to view the details of all the lecturers in his/her department. Human resources staff should access the system through a single user name and password and the department secretaries should have they own different user name and password which is stored with each department.

This technical report details the design, implementation and testing of the system solution. The design section contains the use case and class diagrams crucial to object-orientated principles in design.

# Analysis

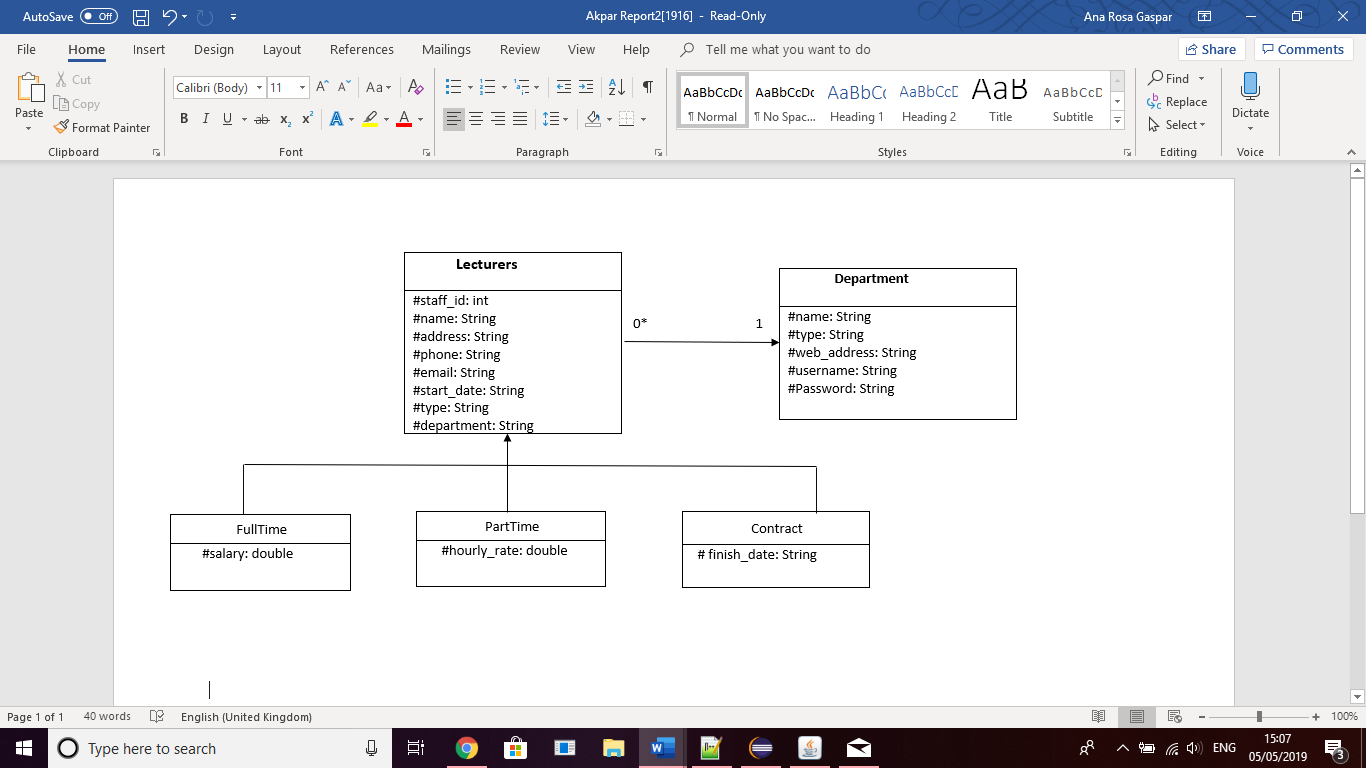
Analyzing the brief provided by the Northampton Metropolitan University it’s possible to find potential objects, get the interactions between them, classifications and internal attributes. The analyze of the brief provides a key insight for the later stage of design.

The brief indicated a variety of candidate classes: Departments, Lectures, Contract, Full-Time, Part-Time. It also suggested several responsibilities for candidate classes, such as: open, view, login, exit, add, remove, edit, query.

Further analysis of the system requirements results in several assumptions about the task, such as the use of class inheritance, abstract classes, Array Lists, generics, file and exception handling.

# Design

## Class Diagram



## Use Case Diagram

A close up of a map

Description automatically generated

Table Use Case Query Lecturers

|  |  |
| --- | --- |
| **Identifier:** | Query Lecturer |
| **Initiator:** | Secretary |
| **Goal:** | Secretary views lecturers from her/his department |
| **Pre-condition:** | Software is loaded with secretary menu |
| **Post-conditions:** | Secretary success views the lectures |
| **Assumptions:** | Secretary is a valid secretary from a register department on the system and logged into the system. |
| **Main success scenario:** | 1. Secretary clicks the view lectures menu item which brings up a panel with a text area. 2. System displays lectures on the text area. |
| **Includes:** | - |
| **Extends:** | - |

Table Use Case Login

|  |  |
| --- | --- |
| **Identifier:** | Login |
| **Initiator:** | Secretary, HR staff |
| **Goal:** | Secretary/HR staff logs on the system |
| **Pre-condition:** | Software is loaded on the login screen |
| **Post-conditions:** | Secretary/HR staff successfully logs on the system |
| **Assumptions:** | Secretary/HR staff is a valid secretary/HR registered within the system. |
| **Main success scenario:** | 1. Secretary/HR staff types their username and password into the systems GUIs respective text fields 2. System compares secretary/HR staff entered username and password against possible registered departments 3. If secretary/HR staff entered username and password are valid then the system grants access to the secretary/HR staff menu bar. |
| **Includes:** | Add Lectures, Edit Lectures, Remove Lectures, View Lectures, View Departments, Remove Departments, Edit Departments, Add Departments, Query Lectures |
| **Extends:** | - |

Table Use Case Add Department

|  |  |
| --- | --- |
| **Identifier:** | Add Department |
| **Initiator:** | HR |
| **Goal:** | HR staff adds new department to system |
| **Pre-condition:** | Software is loaded and HR staff is logged in |
| **Post-conditions:** | HR staff successfully adds a department to the system |
| **Assumptions:** | HR staff is logged into the system with the HR staff username and password. |
| **Main success scenario:** | 1. HR staff clicks add department menu item which brings up a panel with fields. 2. HR staff fill in the panel fields with relevant information about the department. 3. HR staff clicks the submit button which stores the new department information. 4. The new department information is stored within the system. |
| **Includes:** | - |
| **Extends:** | - |

Table Use Case Add Lecturer

|  |  |
| --- | --- |
| **Identifier:** | Add Lecturer |
| **Initiator:** | HR staff |
| **Goal:** | HR staff adds new lecturer to system |
| **Pre-condition:** | Software is loaded, and HR staff is logged in |
| **Post-conditions:** | HR staff successfully adds a lecturer to the system |
| **Assumptions:** | HR staff is logged into the system with the HR staff username and password. |
| **Main success scenario:** | 1. HR staff clicks add lecturer menu item which brings up a panel with fields. 2. HR staff fill in the panel fields with relevant information about the lecturer. 3. HR staff clicks the submit button which stores the new lecturer information. 4. The new lecturer information is stored within the system. |
| **Includes:** | - |
| **Extends:** | - |

Table Use Case Remove Department

|  |  |
| --- | --- |
| **Identifier:** | Remove Department |
| **Initiator:** | HR staff |
| **Goal:** | HR staff removes department from the system |
| **Pre-condition:** | Software is loaded, and HR staff is logged in |
| **Post-conditions:** | HR staff successfully removes a department from the system |
| **Assumptions:** | HR staff is logged into the system with the HR staff username and password. |
| **Main success scenario:** | 1. HR staff clicks edit department menu item which brings up a panel with fields. 2. HR staff selects the name of the department 3. System fill the panel fields with that department information. 4. HR staff clicks the remove button. 5. The department is removed from the system. |
| **Includes:** | - |
| **Extends:** | - |

Table Use Case Remove Lecturer

|  |  |
| --- | --- |
| **Identifier:** | Remove Lecturer |
| **Initiator:** | HR staff |
| **Goal:** | HR staff removes lecturer from the system |
| **Pre-condition:** | Software is loaded, and HR staff is logged in |
| **Post-conditions:** | HR staff successfully removes a lecturer from the system |
| **Assumptions:** | HR staff is logged into the system with the HR staff username and password. |
| **Main success scenario:** | 1. HR staff clicks edit lecturer menu item which brings up a panel with fields. 2. HR staff selects the staff id of the lecturer 3. System fill the panel fields with that lecturer information. 4. HR staff clicks the remove button. 5. The lecturer is removed from the system. |
| **Includes:** | - |
| **Extends:** | - |

Table Use Case Edit Lecturer

|  |  |
| --- | --- |
| **Identifier:** | Edit Lecturer |
| **Initiator:** | HR staff |
| **Goal:** | HR staff edit lecturer on the system |
| **Pre-condition:** | Software is loaded, and HR staff is logged in |
| **Post-conditions:** | HR staff successfully edited a lecturer on the system |
| **Assumptions:** | HR staff is logged into the system with the HR staff username and password. |
| **Main success scenario:** | 1. HR staff clicks edit lecturer menu item which brings up a panel with fields. 2. HR staff selects the staff id of the lecturer 3. System fill the panel fields with that department information. 4. HR staff fill in the panel fields that what to be changed with relevant information about the lecturer. 5. HR staff clicks the edit button. 6. The lecturer information is changed on the system. |
| **Includes:** | - |
| **Extends:** | - |

Table Use Case Edit Department

|  |  |
| --- | --- |
| **Identifier:** | Edit Department |
| **Initiator:** | HR staff |
| **Goal:** | HR staff edit department on the system |
| **Pre-condition:** | Software is loaded, and HR staff is logged in |
| **Post-conditions:** | HR staff successfully edited a department on the system |
| **Assumptions:** | HR staff is logged into the system with the HR staff username and password. |
| **Main success scenario:** | 1. HR staff clicks edit department menu item which brings up a panel with fields. 2. HR staff selects the name of the department 3. System fill the panel fields with that department information. 4. HR staff fill in the panel fields that what to be changed with relevant information about the department. 5. HR staff clicks the edit button. 6. The department information is changed on the system. |
| **Includes:** | - |
| **Extends:** | - |

Table Use Case Query Department

|  |  |
| --- | --- |
| **Identifier:** | Query Department |
| **Initiator:** | HR staff |
| **Goal:** | HR staff views departments from the system |
| **Pre-condition:** | Software is loaded with HR menu |
| **Post-conditions:** | HR staff successfully views the departments |
| **Assumptions:** | HR staff is logged into the system with the HR staff username and password. |
| **Main success scenario:** | 1. HR staff clicks view department menu item which brings up a panel with a text area. 2. System displays departments on the text area. |
| **Includes:** | - |
| **Extends:** | - |

Table Use Case Query Lecturer

|  |  |
| --- | --- |
| **Identifier:** | Query Lecturer |
| **Initiator:** | HR staff |
| **Goal:** | HR staff views lecturer from the system |
| **Pre-condition:** | Software is loaded with HR menu |
| **Post-conditions:** | HR staff successfully views the lectures |
| **Assumptions:** | HR staff is logged into the system with the HR staff username and password. |
| **Main success scenario:** | 1. HR staff clicks view lectures menu item which brings up a panel with a text area. 2. System displays lectures on the text area. |
| **Includes:** | - |
| **Extends:** | - |

## Overall design

The overall design of the software system is simple, employing only five object classes, four concerning lecturers and one being departments. The lecturer object class itself is abstract as it isn’t meant to be initialized itself as it doesn’t make up a whole object, it is only part of other objects requires by the system.

The classes collaborate with one another at some level, most noticeably the objects contract, fulltime and parttime are all subclasses and inherit attributes and methods from the lecturer superclass object. This offers several benefits to the software system by allowing the subclass to all be recognized as there superclass at some level whilst still being distinguishable from one another. It also reduces repeated code between the different subclass objects as they all need to be able store the same attributes, staff\_id, lecturerType, name, address, phone, email, start\_date, department and have access to the same methods regarding those attributes.

## System Description (Main components and functionalities)

The system is comprised of various components and functionalities, designed to facilitate a university’s need to keep tracking of the lectures of different departments. The system allows member of the Human Resources team to:

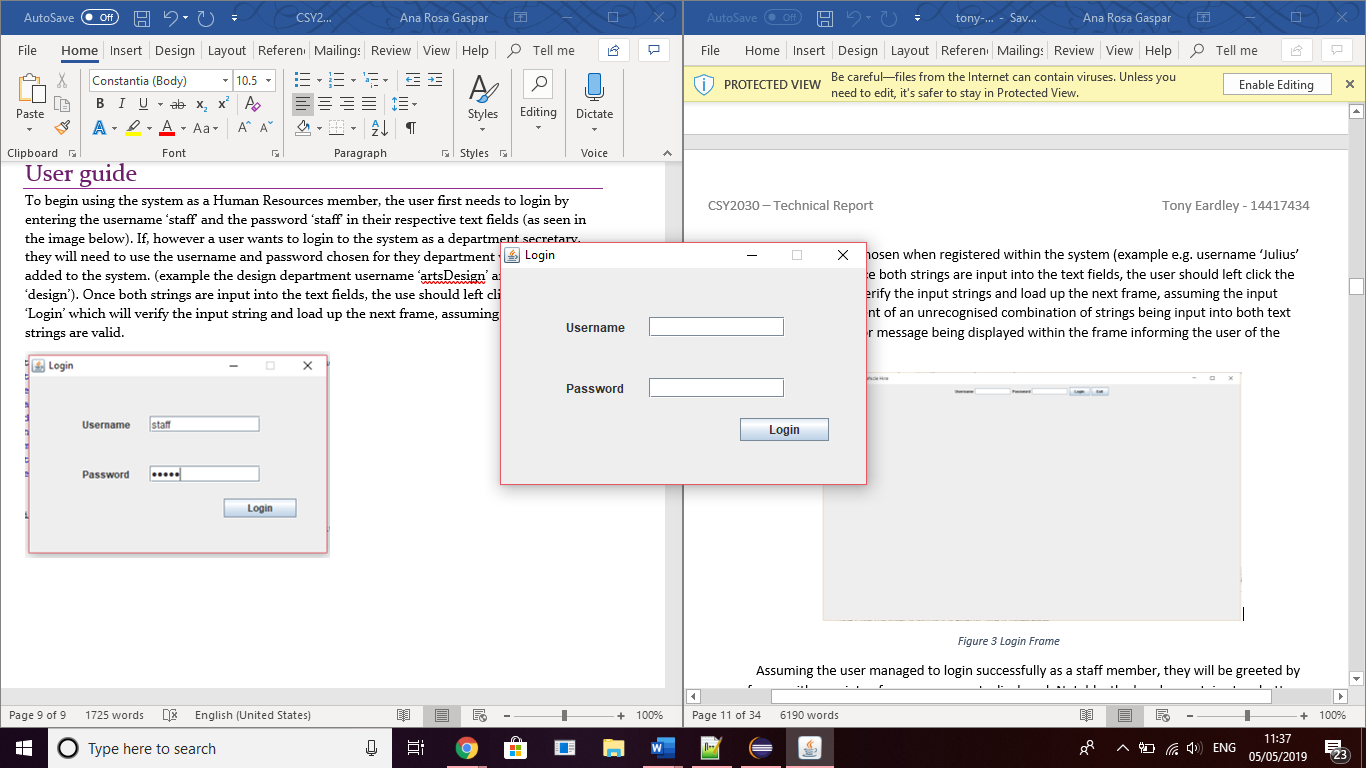
* Log onto the system with the username ‘staff’ and password ‘staff’.
* Exit the system
* Add different types of lectures to the system
* Remove lectures from the system
* Change details of a lecturer
* Add departments to the system
* Change details of a department
* Remove a department
* View all the lectures in the system
* View all the departments in the system

The system also allows department secretaries to:

* Log onto the system with a username and password that is stored in the department
* Exit the system
* Show details of all types of lecturers
* Query a specific lecture’s details by the id (on the edit)

# User guide

To begin using the system as a Human Resources member, the user first needs to login by entering the username ‘staff’ and the password ‘staff’ in their respective text fields (as seen in the image below). If, however a user wants to login to the system as a department secretary, they will need to use the username and password chosen for they department when it was added to the system. (example the design department username ‘artsDesign’ and password ‘design’). Once both strings are input into the text fields, the use should left click the button ‘Login’ which will verify the input string and load up the next frame, assuming the input strings are valid.

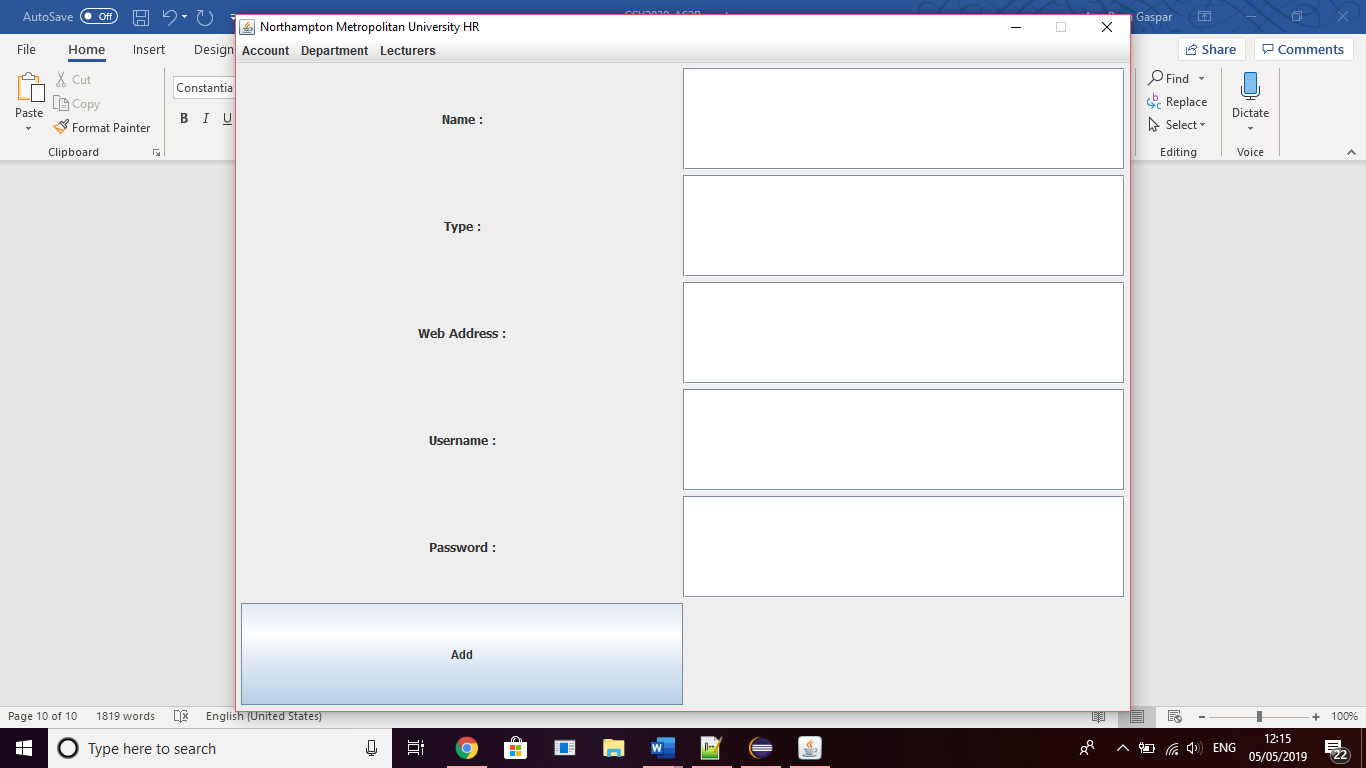
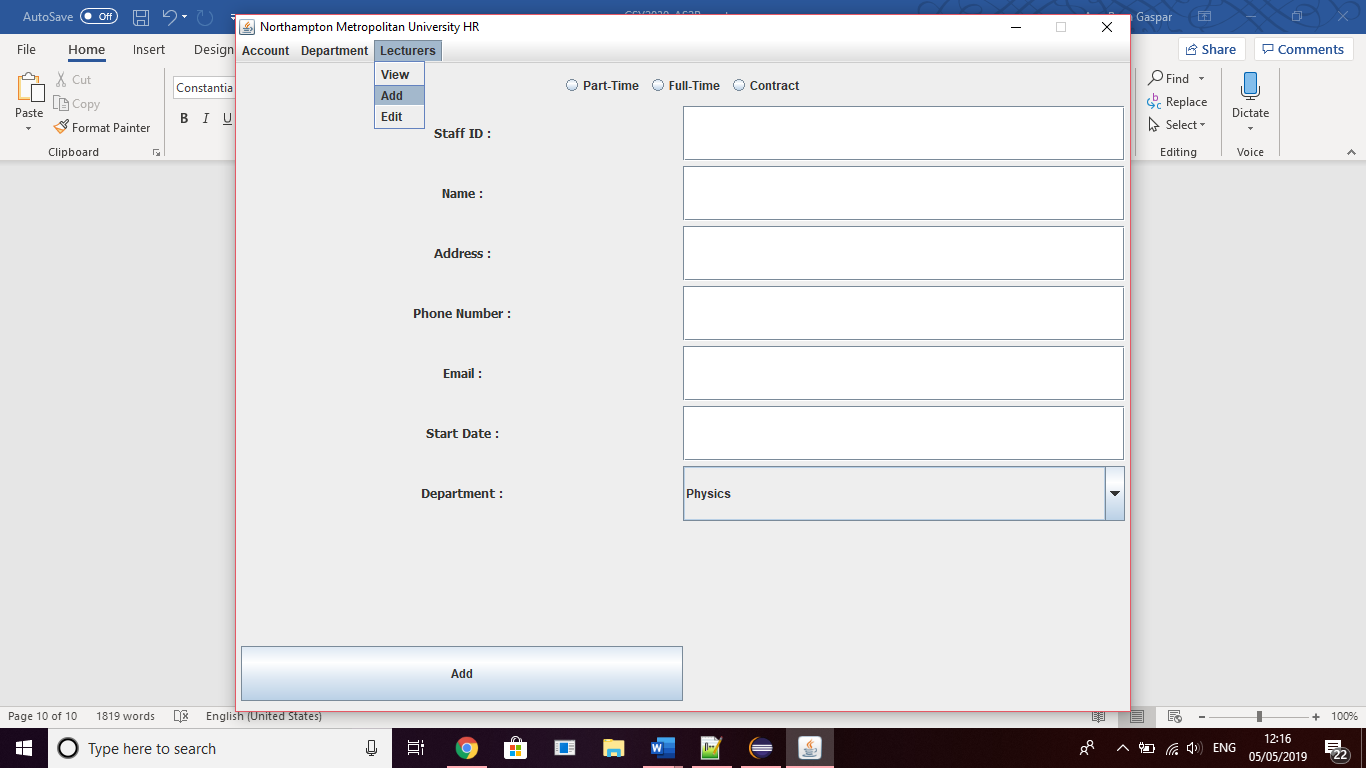


If the user managed to login successfully as a HR member, they will be greeted by a new frame with a variety of new components displayed. Notably, the header whit a menu bar with three menu options ‘Account’, ‘Department’, ‘Lecturers’. These menu items determine what information is being displayed within the frame.



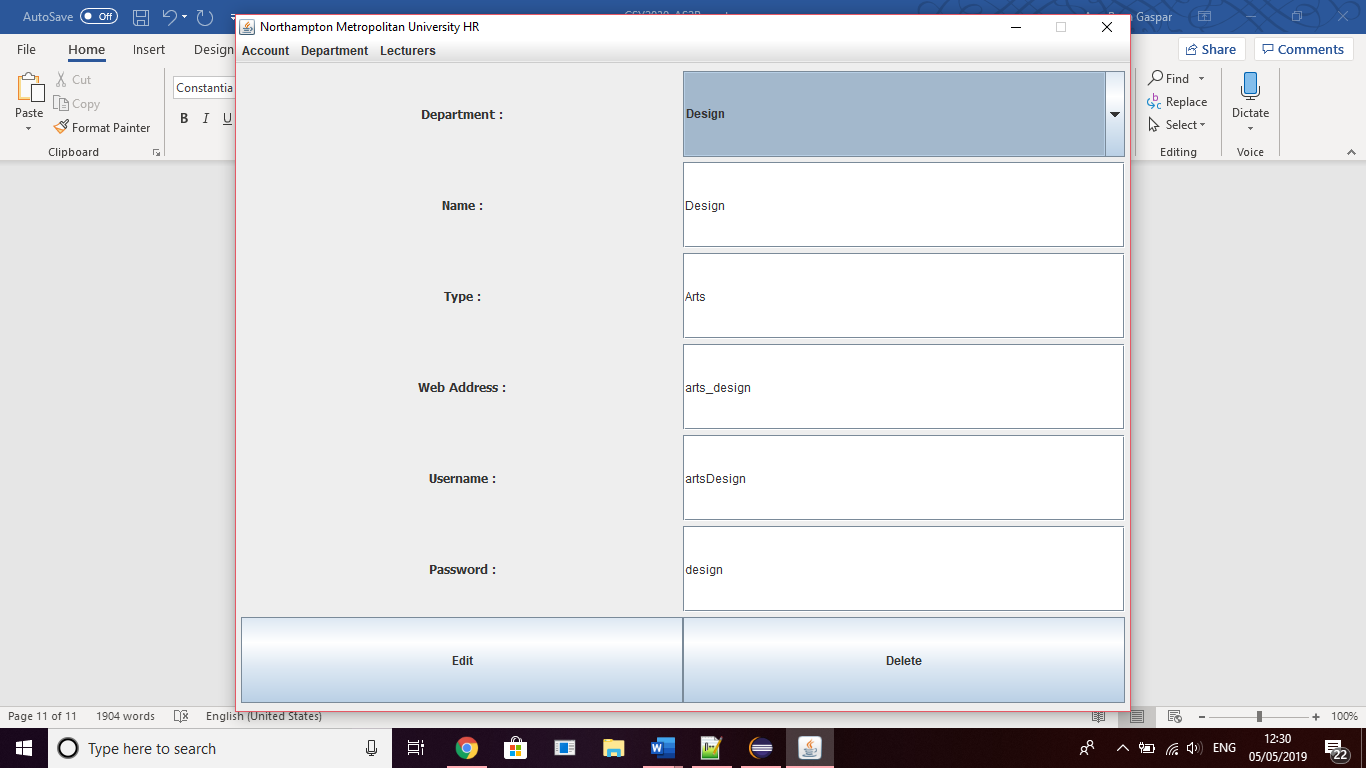
Both the ‘Lecturers’ and ‘Department’ item display a dropdown list that contains the view, add and edit links. This links contains different forms for different proposes.

So, if we want to add a new lecture or department, we select the add item.

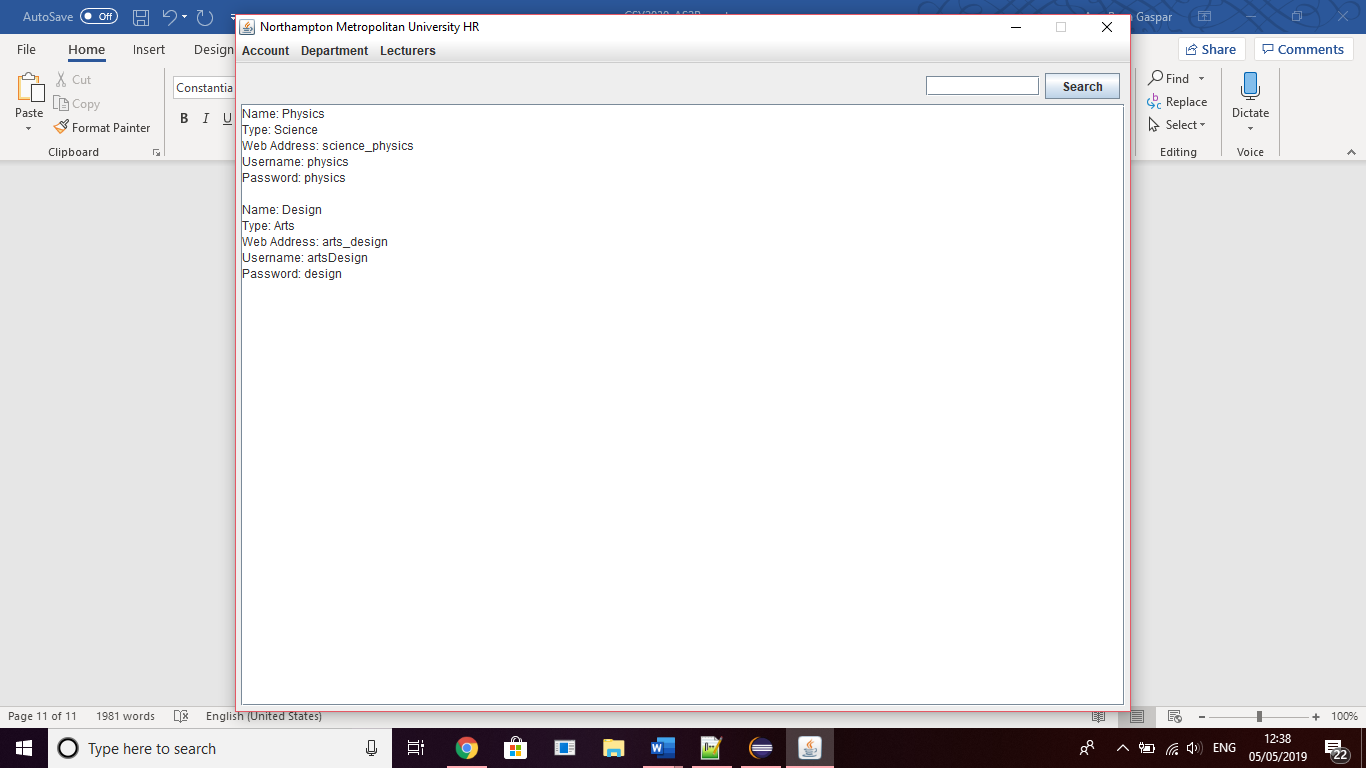
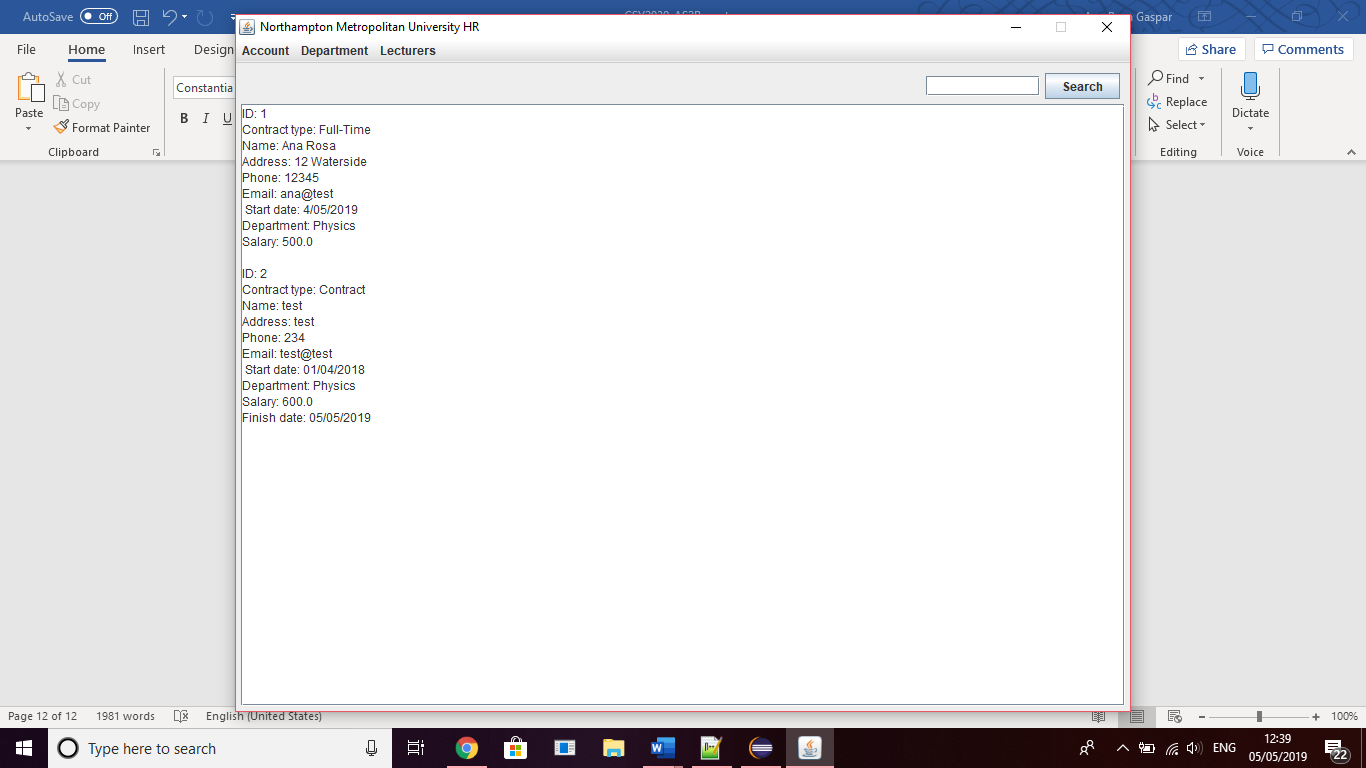


Once we insert the data and click on the add button the panel will refresh and clean all the fields.

To remove or edit the panel is the same. The panel is similar except for the use of combo boxes to get the staff\_id in the case of the lecturer and the name in the department. This combo boxes help us to get all the information of a lecture or department through they id or name. So, if we want to edit the Design department, we only have to select it and all information of that department comes up helping us to edit without errors.



After changing what we wanted to change, we can confirm if the changes happened on the view item. The view is where the files are displayed, if we go to the department view, we can see all the departments in the department files. If we go to the lecturer’s view a list of all lectures is showed.



However, if we are logged in as a department secretary the only menus options that we have are ‘Account’ and ‘Lecturers’. Unlike the HR staff the secretaries are only supposed to see the lecturers view and on they view only they are only able to see the lectures that work on they department.

The ‘Account’ item allows the user to logout and save all alterations they did to lecturers or departments. Although that the system is also really to save everything if the system is closes improperly.

# Testing

|  |  |  |  |
| --- | --- | --- | --- |
| Test | Expected Result | Actual Result | Action/Potential Fix |
| Valid login (HR)  Username ‘staff’  Password ‘staff’ | Login successful – display welcome frame | As expected. | - |
| Valid login (secretary)  Username ‘physics’  Password ‘physics’ | Login successful – display welcome frame with only ‘Accounts’ and ‘Lecturers’ on menu bar | As expected. | - |
| Invalid login (empty text fields) | Console show invalid login error | As expected |  |
| Account -> exit (secretary and HR) | System closes and alterations are saved | As expected | - |
| View lectures (secretary)  Physics department secretary | Only lecturers that are from the physics department | As expected | - |
| Add lecturer    Id : 4  Type: Part-Time  Name: Ana  Address: 15 Waterside  Phone: 3456  Email: test  Start date: 20/05/2019  Department: Design  Hourly rate:45 | Lecturer added | As expected | - |
| Edit lecturer  Change email to ‘ana@test’ where id is 4 | Lecturer changed | As expected | - |
| Remove lecturer  Remove lecture with id = 2 | Lecturer deleted | As expected | Lecture was deleted but is still showing on the edit lecturer combo box |
| Add department  Name: IT  Type: Computing  Web Address: test  Username: ITComputing  Password: computing- | Department added | As expected | - |
| Edit department  Change name to ‘Computing’ where name is IT | Department name edited | As expected | Department was deleted but is still showing on the edit department combo box |
| Delete department  Delete computing department | Department deleted | As expected | - |

# Conclusion

In conclusion, the software system provides an adequate solution to the assignment brief, with all the minimum system requirements and additional system requirements being built into the system. The system utilizes an MVC (Model View Controller) approach, whilst also implementing the use of collections and appropriate exception handling. The system ensures lectures and departments created during a session are saved to a file in the event of a system exit and are reloaded when the program is opened again.

A great addition to software system’s design was the use of generics which reduced repeated code considerably and increased the programs overall efficiency and flexibility as can be seen in the model. One method exists for the following actions; adding objects to any given array list, removing objects from array lists, writing array lists of objects to binary files, reading objects from binary files into array lists, and finding specific objects from any given array lists.

Other additional features which would be beneficial if implemented; a logout functionality as opposed to having to exit the program entirely, error checking being built into the object classes as opposed to the system, this would improve the classes reusability.

# References

JPanel, S. and Muchai, O. (2019). *Search text file and display results in a JPanel*. [online] Stack Overflow. Available at: https://stackoverflow.com/questions/17985808/search-text-file-and-display-results-in-a-jpanel/17988175 [Accessed 5 Mar. 2019].

Docs.oracle.com. (2019). How to Use CardLayout (The Java™ Tutorials > Creating a GUI With JFC/Swing > Laying Out Components Within a Container). [online] Available at: https://docs.oracle.com/javase/tutorial/uiswing/layout/card.html [Accessed 20 Apr. 2019].