**New Zealand Diploma in Information Systems**

**HTCS5607 IS Application Project**

**TECHNICAL REPORT TEMPLATE**

**Project Name:** saint albert

|  |  |  |  |
| --- | --- | --- | --- |
| **Project Team Name(s)** | **Student ID** | **Email** | **Phone** |
| ***Project Manager***  *Jorge Campero* | 1551601 | jorgecamperogarcia@gmail.com | 0224715962 |
| ***Deputy Project Manager***  *Jorge Campero* | 1551601 | jorgecamperogarcia@gmail.com | 0224715962 |
| ***Team Members***  *Jorge Campero* | 1551601 | jorgecamperogarcia@gmail.com | 0224715962 |
|  |  |  |  |
|  |  |  |  |

**Client Stakeholders**

|  |  |  |
| --- | --- | --- |
| **Client Stakeholders** | **Full name and title** | **Contact details** |
| **Project Sponsor(s)** | *Lei Song, Simon Dacey* | *[phone, email]* |
| ***[add other roles as appropriate]*** | *[full name and title]* | *[phone, email]* |

**DATE OF SUBMISSION**

*dd/mm/yyyy*

Table of Contents

[1. Document Control 4](#_Toc80799791)

[1.1 Version History 4](#_Toc80799792)

[1.2 Contribution to Report sections 4](#_Toc80799793)

[1.3 Glossary 4](#_Toc80799794)

[2. Executive Summary 5](#_Toc80799795)

[3. Introduction 6](#_Toc80799796)

[4. Technology Review 7](#_Toc80799797)

[5. IT Methodology 9](#_Toc80799798)

[6. Project Management 10](#_Toc80799799)

[6.1 Project Management Narrative 10](#_Toc80799800)

[6.2 Project Plan with Milestones 10](#_Toc80799801)

[6.3 Project Governance Responsibilities 10](#_Toc80799802)

[6.4 Project Meetings 10](#_Toc80799803)

[6.5 Project Reports 10](#_Toc80799804)

[6.6 Project Risk and Issue Analysis 10](#_Toc80799805)

[7. Requirements Analysis 11](#_Toc80799806)

[7.1 Introduction 11](#_Toc80799807)

[7.2 Use Case Diagram 12](#_Toc80799808)

[7.3 Business Use Case Narratives (Descriptions) 13](#_Toc80799809)

[7.4 Activity Diagrams 27](#_Toc80799810)

[7.5 Overall Class Diagram 35](#_Toc80799811)

[8. Project Design 36](#_Toc80799812)

[8.1 Introduction 36](#_Toc80799813)

[8.2 Software List 36](#_Toc80799814)

[8.3 Version Control Software 36](#_Toc80799815)

[8.4 Design Use Case Narratives (Descriptions) 37](#_Toc80799816)

[8.5 Sequence Diagrams 51](#_Toc80799817)

[8.6 Deployment Diagram 55](#_Toc80799818)

[8.7 Database Design 56](#_Toc80799819)

[8.8 Annotated User Interface Designs 60](#_Toc80799820)

[8.9 Test Plan 65](#_Toc80799821)

[9. Project Training 66](#_Toc80799822)

[9.1 End User Background and Training Objectives 66](#_Toc80799823)

[9.2 Training Materials 66](#_Toc80799824)

[9.3 Training Deliverables 66](#_Toc80799825)

[9.4 Evaluation 66](#_Toc80799826)

[10. Conclusion & Lessons Learned 67](#_Toc80799827)

[References 68](#_Toc80799828)

[Appendices 69](#_Toc80799829)

# 1. Document Control

## 1.1 Version History

This document has had the following revisions:

| **Version** | **Date** | **Author** | **Description of Change** |
| --- | --- | --- | --- |
| 0.1 |  |  | Initial draft |

## 1.2 Contribution to Report sections

| **Project Team Member name** | **Student ID** | **Report Section** |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

## 1.3 Glossary

To provide clarity, terms and acronyms used in this document are defined as follows:

| **Term / Abbreviation** | **Definition** |
| --- | --- |
| Supervisor | Technical Advisor |
|  |  |

# 2. Executive Summary

# 3. Introduction

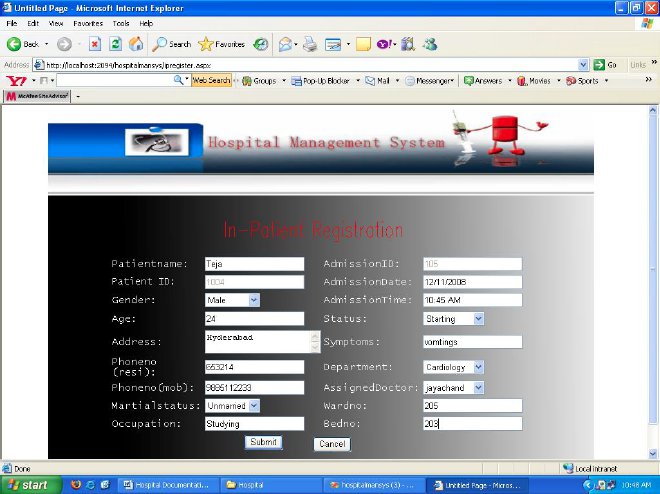
# 4. Technology Review

Some of the modern GUI languages are C#, Python, C++, and Java. C# is mainly used for windows applications, while Python, C++ and Java are cross-platform. These languages have different frameworks that helps with the development of the user interface, for instance Python has Tkinter, PyQt5 and Kivy, each of these frameworks have its advantages when developing an UI, being Tkinter the most popular one due to its level of customization. For C++ we can find Qt, wxWidgets, JUCE, CEGUI, and CEF. For Java we can use AWT, Swing and JavaFX, being JavaFX the most popular one nowadays, while AWT and Swing are more of a legacy framework. For this project, I need to use one language that offers a balance between functionality and design, in this case C# is the best one as it offers great possibilities for design.

We have also to consider the IDE (integrated development environment) of each language, for C# we have Visual Studio which is without a doubt the best IDE for this language, its grab and drop system for the GUI part makes Visual Studio easy to use. For Python we have PyCharm which is perfect for testing and for uploading the code to GitHub. And for the other languages Visual Studio Code can be used as it is relatively easy to use and support several programming languages. As I said before, both Python and C# have great IDEs but for the purpose of this project, Visual Studio is the best because for the development of the user interface it offers the option of grabbing the objects and paste them on the form.

For storing the data, we have several options such as MySQL, Microsoft Access, and OpenOffice Base, which are all local databases. For server like databases, we have Azure MySQL, Amazon Aurora, and MongoDB Cloud. For this project, Microsoft Access is the best as it meets the requirements, and it is compatible with C#.

As for similar projects done in the past, we have this project which have similar use cases as the ones describe in the Saint Albert project. Here are some screenshots from that project.



# 5. IT Methodology

There are several types of development methodologies such as Agile and Waterfall, each of those have their advantages and disadvantages. Some of the advantages of the Agile methodology is that it is quicker and allows for more flexibility in the development of the project, while the Waterfall methodology is that structure is clear from the start, and it allows the team to plan for a specific timeline. It is better suit for small projects. For this project I have chosen the waterfall methodology as the project is small and the goals are clear.

The phases of the waterfall methodology are Information Gathering/Requirements, Analysis, Design, Development, Deployment and Maintenance.

# 6. Project Management

## 6.1 Project Management Narrative

*Details with evidence how the development of the project followed the selected systems development lifecycle*

## 6.2 Project Plan with Milestones

*Include an overall plan here and attach a detailed GANTT chart to the appendices*

## 6.3 Project Governance Responsibilities

*Explain who was responsible for project management and quality assurance, and explain how these tasks were carried out*

## 6.4 Project Meetings

*Include a schedule of your meetings (date, duration, participants, and type) and attach the minutes of each meeting to the appendices*

## 6.5 Project Reports

*Discuss the project status reports and attach your project status reports to the appendices*

## 6.6 Project Risk and Issue Analysis

*Discuss project risks and issues and attach your project risk and issue register to the appendices*

# 7. Requirements Analysis

## 7.1 Introduction

## 7.2 Use Case Diagram

Diagram

Description automatically generated

## 7.3 Business Use Case Narratives (Descriptions)

|  |  |  |
| --- | --- | --- |
| **USE CASE NAME:** | Add Admission | **USE CASE TYPE** |
| **USE CASE ID:** | 10 | **Business Requirements: 🗹** |
| **PRIORITY:** | High |  |
| **PRIMARY BUSINESS ACTOR:** | Assistant Administrator | |
| **OTHER PARTICIPATING ACTORS:** | None. | |
| **DESCRIPTION:** | This use case enables the assistant administrator to add a selected patient’s admission details. | |
| **PRE-CONDITIONS:** | The assistant administrator has logged onto the system | |
| **TYPICAL COURSE** | Step 1- The assistant administrator selects the “Add Admission” function.  Step 2- The system displays the “Add Admission” form with all fields blank.  Step 3- The assistant administrator enters the admission’s details (admission description and admission date).  Step 4- The system checks that the details are filled in correctly.  Step 5- The system displays a list of patients (patient id, last name, and first name).  Step 6- The assistant administrator selects a patient.  Step 7- The system displays a list of wards (ward id and ward name).  Step 8- The assistant administrator selects a ward.  Step 9- The assistant administrator elects to add the admission.  Step 10- The system checks that the details are filled in correctly.  Step 11- The system saves the admission’s details.  Step 12- The system displays the message “Admission added successfully”.  Step 13- The system displays the “Add another admission?” prompt.  Step 14 - The assistant administrator elects to close the form.  Step 15- The system ends the use case. | |
| **OF EVENTS:** |
|  |
|  |
|  |
|  |
| **ALTERNATE COURSES:** | Step 4A1- The system identifies that some details are missing.  Step 4A2- The system prompts for the completion of the details. | |
| Step 9A1- The assistant administrator elects to cancel.  Step 9A2- The system closes the form. | |
|  | Step 14A1- The assistant administrator elects to add another admission.  Step 14A2- The system goes to step 2. | |
| **POST CONDITIONS:** | None. | |
| **ASSUMPTIONS:** | None. | |

|  |  |  |
| --- | --- | --- |
| **USE CASE NAME:** | Update Admission | **USE CASE TYPE** |
| **USE CASE ID:** | 11 | **Business Requirements: 🗹** |
| **PRIORITY:** | High |  |
| **PRIMARY BUSINESS ACTOR:** | Assistant Administrator | |
| **OTHER PARTICIPATING ACTORS:** | None. | |
| **DESCRIPTION:** | This use case enables the assistant administrator to update a selected admission’s details. | |
| **PRE-CONDITIONS:** | The assistant administrator has logged onto the system | |
| **TYPICAL COURSE** | Step 1- The assistant administrator selects the “Update Admission” function.  Step 2- The system displays the “Update Admission” form with a list of all the current admissions (admission id and description).  Step 3- The assistant administrator selects the admission that has details that needs updating.  Step 4- The system displays the admission’s details (admission id, description, admission date, status, patient last name, patient first name, and ward name).  Step 5- The assistant administrator updates the relevant details (description, status (either current or complete only), and admission date only).  Step 6- The system validates the entries in the fields.  Step 7- The system prompts for confirmation to change the admission’s details.  Step 8- The assistant administrator confirms the change of details.  Step 9- The system saves the admission’s details.  Step 10- The system displays the message “Admission updated successfully”.  Step 11- The system displays the “Update another admission?” prompt.  Step 12- The assistant administrator elects to close the form.  Step 13- The system ends the use case. | |
| **OF EVENTS:** |
|  |
|  |
|  |
|  |
| **ALTERNATE COURSES:** | Step 3A1- The assistant administrator elects to cancel the operation.  Step 3A2- The system closes the form. | |
| Step 7A1- The system identifies missing or incorrect fields.  Step 7A2- The system prompts for completion of the entry. | |
|  | Step 8A1- The assistant administrator elects to cancel the changes.  Step 8A2- The system closes the form. | |
|  | Step 12A1- The assistant administrator elects to update another admission.  Step 12A2- The system goes to step 2. | |
| **POST CONDITIONS:** | None. | |
| **ASSUMPTIONS:** | None. | |

|  |  |  |
| --- | --- | --- |
| **USE CASE NAME:** | Delete Admission | **USE CASE TYPE** |
| **USE CASE ID:** | 12 | **Business Requirements: 🗹** |
| **PRIORITY:** | High |  |
| **PRIMARY BUSINESS ACTOR:** | Assistant Administrator | |
| **OTHER PARTICIPATING ACTORS:** | None. | |
| **DESCRIPTION:** | This use case enables the assistant administrator to delete a selected admission’s details. | |
| **PRE-CONDITIONS:** | The assistant administrator has logged onto the system | |
| **TYPICAL COURSE** | Step 1- The assistant administrator selects the “Delete Admission” function.  Step 2- The system displays the “Delete Admission” form with a list of all the closed admissions (admission id and description).  Step 3- The assistant administrator selects the admission that requires deleting.  Step 4- The system displays the admission’s details (admission id, description, admission date, and status).  Step 5- The assistant administrator elects to delete the admission.  Step 6- The system deletes all payments associated with the admission.  Step 7- The system deletes the admission.  Step 8- The system displays the message “Admission deleted successfully”.  Step 9- The system displays the “Delete another admission?” prompt.  Step 10- The assistant administrator elects to close the form.  Step 11- The system ends the use case. | |
| **OF EVENTS:** |
|  |
|  |
|  |
|  |
| **ALTERNATE COURSES:** | Step 3A1- The assistant administrator elects to cancel the operation.  Step 3A2- the system closes the form. | |
| Step 5A1- The assistant administrator elects to cancel the operation.  Step 5A2- the system closes the form. | |
|  | Step 10A1- The assistant administrator elects to delete another admission.  Step 10A2- The system goes to step 2. | |
| **POST CONDITIONS:** | None. | |
| **ASSUMPTIONS:** | None. | |

|  |  |  |
| --- | --- | --- |
| **USE CASE NAME:** | Produce Admissions Report | **USE CASE TYPE** |
| **USE CASE ID:** | 13 | **Business Requirements: 🗹** |
| **PRIORITY:** | High |  |
| **PRIMARY BUSINESS ACTOR:** | The assistant administrator | |
| **OTHER PARTICIPATING ACTORS:** | None. | |
| **DESCRIPTION:** | This use case enables the assistant administrator to produce the admissions report. | |
| **PRE-CONDITIONS:** | The assistant administrator has logged onto the system | |
| **TYPICAL COURSE** | Step 1- The assistant administrator selects the “Admissions Report” function.  Step 2- The system displays the “Admissions Report” form.  Step 3- The assistant administrator selects the option to generate the report.  Step 4- The system gets the details (admission ID, description, admission date, and status) of each admission.  Step 5- The system gets the patient’s last name and first name for each admission.  Step 6- The system gets the name of each medication prescribed to each admission.  Step 7- The system then displays the admissions report (admission ID, description, admission date, status, patient last name, patient first name, and medication names for each admission).  Step 8- The system closes the form.  Step 9- The system ends the use case. | |
| **OF EVENTS:** |
|  |
|  |
|  |
|  |
| **ALTERNATE COURSES:** | Step 3A1- The assistant administrator elects to cancel without generating the report.  Step 3A2- The system closes the form. | |
| **POST CONDITIONS:** | None | |
| **ASSUMPTIONS:** | None | |

|  |  |  |
| --- | --- | --- |
| **USE CASE NAME:** | Remove Prescription | **USE CASE TYPE** |
| **USE CASE ID:** | 15 | **Business Requirements: 🗹** |
| **PRIORITY:** | High |  |
| **PRIMARY BUSINESS ACTOR:** | Pharmacy Administrator | |
| **OTHER PARTICIPATING ACTORS:** | None | |
| **DESCRIPTION:** | This use case enables the pharmacy administrator to remove a prescription from a selected admission. | |
| **PRE-CONDITIONS:** | The pharmacy administrator has logged onto the system. | |
| **TYPICAL COURSE** | Step 1- The pharmacy administrator selects the “Remove Prescription” function.  Step 2- The system displays the “Remove Prescription” form with a list of all the current admissions (admission id and description) that have prescriptions.  Step 3- The pharmacy administrator selects the admission to remove the prescription from  Step 4- The system displays the admission’s details (admission id, description, and patient’s last name and first name).  Step 5- The system displays a list of the prescriptions (medication name, prescription date, and amount) prescribed to the admission.  Step 6- The pharmacy administrator selects a prescription.  Step 7- The pharmacy administrator elects to remove the prescription from the admission.  Step 8- The system deletes the prescription details.  Step 9- The system displays the message “Prescription removed successfully”.  Step 10- The system displays the “Remove another prescription?” prompt.  Step 11- The pharmacy administrator elects to close the form.  Step 12- The system ends the use case. | |
| **OF EVENTS:** |
|  |
|  |
|  |
|  |
| **ALTERNATE COURSES:** | Step 3A1- The pharmacy administrator elects to cancel the operation.  Step 3A2- The system closes the form. | |
| Step 7A1- The pharmacy administrator elects to cancel the operation.  Step 7A2- The system closes the form. | |
|  | Step 11A1- The pharmacy administrator elects to remove another prescription  Step 11A2- The system goes to step 2. | |
| **POST CONDITIONS:** | None. | |
| **ASSUMPTIONS:** | None. | |

|  |  |  |
| --- | --- | --- |
| **USE CASE NAME:** | Add Research Project | **USE CASE TYPE** |
| **USE CASE ID:** | 27 | **Business Requirements: 🗹** |
| **PRIORITY:** | High |  |
| **PRIMARY BUSINESS ACTOR:** | Research Administrator | |
| **OTHER PARTICIPATING ACTORS:** | None. | |
| **DESCRIPTION:** | This use case enables the research administrator to add a research to a selected doctor. | |
| **PRE-CONDITIONS:** | The research administrator has logged onto the system | |
| **TYPICAL COURSE** | Step 1- The research administrator selects the “Add Research Project” function.  Step 2- The system displays the “Add Research Project” form with a list of all the doctors (doctor id, last name, and first name).  Step 3- The research administrator selects the doctor to add the research project to.  Step 4- The system displays the doctor’s details (doctor id, last name, first name, and specialty).  Step 5- The system displays the research projects’ details (outcome, budget, and research topic description) for each research project already linked to the selected doctor.  Step 6- The system displays a list of the research topic (research topic id, description, and level).  Step 7- The research administrator selects a research topic.  Step 8- The research administrator enters the research project’s outcome, end date and budget.  Step 9- The research administrator elects to add the research project.  Step 10- The system saves the research project details (research project id, doctor, research topic, outcome, end date, and budget).  Step 11- The system displays the “Research project added successfully” message.  Step 12- The system displays the “Add another research project?” prompt.  Step 13- The research administrator elects to close the form.  Step 14- The system ends the use case. | |
| **OF EVENTS:** |
|  |
|  |
|  |
|  |
| **ALTERNATE COURSES:** | Step 3A1- The research administrator elects to cancel the operation.  Step 3A2- The system closes the form. | |
| Step 9A1- The research administrator elects to cancel the operation.  Step 9A2- The system closes the form. | |
|  | Step 13A1- The research administrator elects to add another research project.  Step 13A2- The system goes to step 2. | |
| **POST CONDITIONS:** | None | |
| **ASSUMPTIONS:** | None | |

|  |  |  |
| --- | --- | --- |
| **USE CASE NAME:** | Remove Research Project | **USE CASE TYPE** |
| **USE CASE ID:** | 28 | **Business Requirements: 🗹** |
| **PRIORITY:** | High |  |
| **PRIMARY BUSINESS ACTOR:** | Research Project | |
| **OTHER PARTICIPATING ACTORS:** | None | |
| **DESCRIPTION:** | This use case enables the research administrator to remove a research from selected doctor. | |
| **PRE-CONDITIONS:** | The research administrator has logged onto the system. | |
| **TYPICAL COURSE** | Step 1- The research administrator selects the “Remove Research Project” function.  Step 2- The system displays the “Remove Research Project” form with a list of all the doctors (doctor id, last name, and first name) who have research projects.  Step 3- The research administrator selects the doctor to remove the research project from.  Step 4- The system displays the doctor’s details (doctor id, last name, first name, and specialty).  Step 5- The system displays the research projects’ details (outcome, budget, and research topic description) for each research project linked to the selected doctor.  Step 6- The research administrator selects the research project to remove.  Step 7- The research administrator elects to remove the research project.  Step 8- The system deletes the research project’s details.  Step 9- The system displays the message “Research project removed successfully”.  Step 10- The system displays the “Remove another research project?” prompt.  Step 11- The research administrator elects to close the form.  Step 12- The system ends the use case. | |
| **OF EVENTS:** |
|  |
|  |
|  |
|  |
| **ALTERNATE COURSES:** | Step 3A1- The research administrator elects to cancel the operation.  Step 3A2- The system closes the form. | |
| Step 7A1- The research administrator elects to cancel the operation.  Step 7A2- The system closes the form. | |
|  | Step 13A1- The research administrator elects to remove another research project.  Step 13A2- The system goes to step 2. | |
| **POST CONDITIONS:** | None | |
| **ASSUMPTIONS:** | None | |

|  |  |  |
| --- | --- | --- |
| **USE CASE NAME:** | Produce Research Projects Report | **USE CASE TYPE** |
| **USE CASE ID:** | 32 | **Business Requirements: 🗹** |
| **PRIORITY:** | High |  |
| **PRIMARY BUSINESS ACTOR:** | Research Administrator | |
| **OTHER PARTICIPATING ACTORS:** | None | |
| **DESCRIPTION:** | This use case enables the research administrator to produce the research projects report. | |
| **PRE-CONDITIONS:** | The research administrator has logged onto the system. | |
| **TYPICAL COURSE** | Step 1- The research administrator selects the “Produce Research Projects Report” function.  Step 2- The system displays the “Research Projects Report” form.  Step 3- The assistant administrator selects the option to generate the report.  Step 4- The system gets the details (research project id, outcome, budget, end date, doctor’s id, last name, first name, and research topic description) of each research project.  Step 5- The system then displays the wards report (research project id, outcome, budget, end date, doctor’s id, last name, first name, and research topic description of each research project.) sorted by ward research project id.  Step 6- The system closes the form.  Step 7- The system ends the use case. | |
| **OF EVENTS:** |
|  |
|  |
|  |
|  |
| **ALTERNATE COURSES:** | Step 3A1- The research administrator elects to cancel, without generating the report.  Step 3A2- The system closes the form. | |
| **POST CONDITIONS:** | None | |
| **ASSUMPTIONS:** | None | |

## 7.4 Activity Diagrams*Diagram, schematic Description automatically generatedDiagram Description automatically generatedDiagram Description automatically generatedDiagram Description automatically generatedDiagram Description automatically generatedDiagram Description automatically generatedDiagram Description automatically generatedDiagram Description automatically generated*

## 

## 7.5 Overall Class Diagram

Diagram

Description automatically generated

# 8. Project Design

## 8.1 Introduction

## 8.2 Software List

## 8.3 Version Control Software

## 8.4 Design Use Case Narratives (Descriptions)

|  |  |  |
| --- | --- | --- |
| **USE CASE NAME:** | Add Admission | **USE CASE TYPE** |
| **USE CASE ID:** | 10 | **Design Requirements: 🗹** |
| **PRIORITY:** | High |  |
| **PRIMARY BUSINESS ACTOR:** | Assistant Administrator | |
| **OTHER PARTICIPATING ACTORS:** | None. | |
| **DESCRIPTION:** | This use case enables the assistant administrator to add a selected patient’s admission details. | |
| **PRE-CONDITIONS:** | The assistant administrator has logged onto the system | |
| **TYPICAL COURSE** | Step 1- The assistant administrator selects the “Add Admission” function.  Step 2- The system displays the “Add Admission” form with all fields blank.  Step 3- The assistant administrator enters the admission’s details (admission description and admission date).  Step 4- The system checks that the details are filled in correctly.  Step 5- The system displays a list of patients (patient id, last name, and first name) in a combo box.  Step 6- The assistant administrator selects a patient.  Step 7- The system displays a list of wards (ward id and ward name) in a combo box.  Step 8- The assistant administrator selects a ward.  Step 9- The assistant administrator clicks on the “add admission” button.  Step 10- The system checks that the details are filled in correctly.  Step 11- The system saves the admission’s details.  Step 12- The system displays the message “Admission added successfully”.  Step 13- The system displays the “Add another admission?” prompt.  Step 14 - The assistant administrator clicks on the “Return” button.  Step 15- The system ends the use case. | |
| **OF EVENTS:** |
|  |
|  |
|  |
|  |
| **ALTERNATE COURSES:** | Step 5A1- The system identifies that some details are missing or incorrect, displays the “Please fill in all fields correctly” message.  Step 5A2- The system returns to step 3. | |
| Step 9A1- The assistant administrator clicks on the “Return button.  Step 9A2- The system closes the form. | |
|  | Step 14A1- The assistant administrator elects to add another admission.  Step 14A2- The system goes to step 2. | |
| **POST CONDITIONS:** | None. | |
| **ASSUMPTIONS:** | None. | |

|  |  |  |
| --- | --- | --- |
| **USE CASE NAME:** | Update Admission | **USE CASE TYPE** |
| **USE CASE ID:** | 11 | **Design Requirements: 🗹** |
| **PRIORITY:** | High |  |
| **PRIMARY BUSINESS ACTOR:** | Assistant Administrator | |
| **OTHER PARTICIPATING ACTORS:** | None. | |
| **DESCRIPTION:** | This use case enables the assistant administrator to update a selected admission’s details. | |
| **PRE-CONDITIONS:** | The assistant administrator has logged onto the system | |
| **TYPICAL COURSE** | Step 1- The assistant administrator selects the “Update Admission” function.  Step 2- The system displays the “Update Admission” form with a list of all the current admissions (admission id and description) in a combo box.  Step 3- The assistant administrator selects the admission that has details that needs updating.  Step 4- The system displays the admission’s details (admission id, description, admission date, status, patient last name, patient first name, and ward name) in text boxes.  Step 5- The assistant administrator updates the relevant details (description, status (either current or complete only), and admission date only).  Step 6- The assistant administrator clicks the “Update Admission” button.  Step 7- The system validates the entries in the fields.  Step 8- The system displays the message “Are you sure that you want to update this admission?”.  Step 9- The assistant administrator clicks “Yes”.  Step 10- The system saves the admission’s details.  Step 11- The system displays the message “Admission updated successfully”.  Step 12- The system displays the “Update another admission?” prompt.  Step 13- The assistant administrator clicks the “Return” button.  Step 14- The system ends the use case. | |
| **OF EVENTS:** |
|  |
|  |
|  |
|  |
| **ALTERNATE COURSES:** | Step 3A1- The assistant administrator clicks on the “Return” button.  Step 3A2- The system closes the form. | |
| Step 8A1- The system identifies that some details are missing or incorrect, displays the “Please fill in all fields correctly” message.  Step 8A2- The system returns to step 5. | |
|  | Step 9A1- The assistant administrator clicks “No”.  Step 9A2- The system closes the form. | |
|  | Step 13A1- The assistant administrator elects to update another admission.  Step 13A2- The system goes to step 2. | |
| **POST CONDITIONS:** | None. | |
| **ASSUMPTIONS:** | None. | |

|  |  |  |
| --- | --- | --- |
| **USE CASE NAME:** | Delete Admission | **USE CASE TYPE** |
| **USE CASE ID:** | 12 | **Design Requirements: 🗹** |
| **PRIORITY:** | High |  |
| **PRIMARY BUSINESS ACTOR:** | Assistant Administrator | |
| **OTHER PARTICIPATING ACTORS:** | None. | |
| **DESCRIPTION:** | This use case enables the assistant administrator to delete a selected admission’s details. | |
| **PRE-CONDITIONS:** | The assistant administrator has logged onto the system | |
| **TYPICAL COURSE** | Step 1- The assistant administrator selects the “Delete Admission” function.  Step 2- The system displays the “Delete Admission” form with a list of all the closed admissions (admission id and description) in a combo box.  Step 3- The assistant administrator selects the admission that requires deleting.  Step 4- The system displays the admission’s details (admission id, description, admission date, and status) in text boxes.  Step 5- The assistant administrator clicks on the “Delete Admission” button.  Step 6- The system deletes all payments associated with the admission.  Step 7- The system deletes the admission.  Step 8- The system displays the message “Admission deleted successfully”.  Step 9- The system displays the “Delete another admission?” prompt.  Step 10- The assistant administrator clicks on the “Return” button.  Step 11- The system ends the use case. | |
| **OF EVENTS:** |
|  |
|  |
|  |
|  |
| **ALTERNATE COURSES:** | Step 3A1- The assistant administrator clicks on the “Return” button.  Step 3A2- the system closes the form. | |
| Step 5A1- The assistant administrator clicks on the “Return” button.  Step 5A2- the system closes the form. | |
|  | Step 10A1- The assistant administrator elects to delete another admission.  Step 10A2- The system goes to step 2. | |
| **POST CONDITIONS:** | None. | |
| **ASSUMPTIONS:** | None. | |

|  |  |  |
| --- | --- | --- |
| **USE CASE NAME:** | Produce Admissions Report | **USE CASE TYPE** |
| **USE CASE ID:** | 13 | **Design Requirements: 🗹** |
| **PRIORITY:** | High |  |
| **PRIMARY BUSINESS ACTOR:** | The assistant administrator | |
| **OTHER PARTICIPATING ACTORS:** | None. | |
| **DESCRIPTION:** | This use case enables the assistant administrator to produce the admissions report. | |
| **PRE-CONDITIONS:** | The assistant administrator has logged onto the system | |
| **TYPICAL COURSE** | Step 1- The assistant administrator selects the “Admissions Report” function.  Step 2- The system displays the “Admissions Report” form.  Step 3- The assistant administrator clicks on the “Generate report” button.  Step 4- The system gets the details (admission ID, description, admission date, and status) of each admission.  Step 5- The system gets the patient’s last name and first name for each admission.  Step 6- The system gets the name of each medication prescribed to each admission.  Step 7- The system then displays the admissions report (admission ID, description, admission date, status, patient last name, patient first name, and medication names for each admission).  Step 8- The assistant administrator clicks on the “Return” button.  Step 9- The system closes the form to end the use case. | |
| **OF EVENTS:** |
|  |
|  |
|  |
|  |
| **ALTERNATE COURSES:** | Step 3A1- The assistant administrator clicks on the “Return” button.  Step 3A2- The system closes the form. | |
| **POST CONDITIONS:** | None | |
| **ASSUMPTIONS:** | None | |

|  |  |  |
| --- | --- | --- |
| **USE CASE NAME:** | Remove Prescription | **USE CASE TYPE** |
| **USE CASE ID:** | 15 | **Design Requirements: 🗹** |
| **PRIORITY:** | High |  |
| **PRIMARY BUSINESS ACTOR:** | Pharmacy Administrator | |
| **OTHER PARTICIPATING ACTORS:** | None | |
| **DESCRIPTION:** | This use case enables the pharmacy administrator to remove a prescription from a selected admission. | |
| **PRE-CONDITIONS:** | The pharmacy administrator has logged onto the system. | |
| **TYPICAL COURSE** | Step 1- The pharmacy administrator selects the “Remove Prescription” function.  Step 2- The system displays the “Remove Prescription” form with a list of all the current admissions (admission id and description) that have prescriptions in a combo box.  Step 3- The pharmacy administrator selects the admission to remove the prescription from.  Step 4- The system displays the admission’s details (admission id, description, and patient’s last name and first name) in text boxes.  Step 5- The system displays a list of the prescriptions (prescriptionID, medication Name, prescription date, and amount) prescribed to the admission in a list box.  Step 6- The pharmacy administrator selects a prescription.  Step 7- The pharmacy administrator clicks on the “Remove prescription” button.  Step 8- The system deletes the prescription details.  Step 9- The system displays the message “Prescription removed successfully”.  Step 10- The system displays the “Remove another prescription?” prompt.  Step 11- The pharmacy administrator clicks on the “Return” button.  Step 12- The system ends the use case. | |
| **OF EVENTS:** |
|  |
|  |
|  |
|  |
| **ALTERNATE COURSES:** | Step 3A1- The pharmacy administrator clicks on the “Return” button.  Step 3A2- The system closes the form. | |
| Step 7A1- The pharmacy administrator clicks on the “Return” button.  Step 7A2- The system closes the form. | |
|  | Step 11A1- The pharmacy administrator elects to remove another prescription  Step 11A2- The system goes to step 2. | |
| **POST CONDITIONS:** | None. | |
| **ASSUMPTIONS:** | None. | |

|  |  |  |
| --- | --- | --- |
| **USE CASE NAME:** | Add Research Project | **USE CASE TYPE** |
| **USE CASE ID:** | 27 | **Design Requirements: 🗹** |
| **PRIORITY:** | High |  |
| **PRIMARY BUSINESS ACTOR:** | Research Administrator | |
| **OTHER PARTICIPATING ACTORS:** | None. | |
| **DESCRIPTION:** | This use case enables the research administrator to add research to a selected doctor. | |
| **PRE-CONDITIONS:** | The research administrator has logged onto the system | |
| **TYPICAL COURSE** | Step 1- The research administrator selects the “Add Research Project” function.  Step 2- The system displays the “Add Research Project” form with a list of all the doctors (doctor id, last name, and first name) in a combo box.  Step 3- The research administrator selects the doctor to add the research project to.  Step 4- The system displays the doctor’s details (doctor id, last name, first name, and specialty) in text boxes.  Step 5- The system displays the research projects’ details (outcome, budget, and research topic description) for each research project already linked to the selected doctor in data grid view.  Step 6- The system displays a list of the research topic (research topic id, description, and level) in a combo box.  Step 7- The research administrator selects a research topic.  Step 8- The research administrator enters the research project’s outcome, end date and budget.  Step 9- The research administrator clicks on the “Add Research” button.  Step 10- The system saves the research project details (research project id, doctor, research topic, outcome, end date, and budget).  Step 11- The system displays the “Research project added successfully” message.  Step 12- The system displays the “Add another research project?” prompt.  Step 13- The research administrator clicks on the “Return” button.  Step 14- The system ends the use case. | |
| **OF EVENTS:** |
|  |
|  |
|  |
|  |
| **ALTERNATE COURSES:** | Step 3A1- The research administrator clicks on the “Return” button.  Step 3A2- The system closes the form. | |
| Step 9A1- The research administrator clicks on the “Return” button.  Step 9A2- The system closes the form. | |
|  | Step 13A1- The research administrator elects to add another research project.  Step 13A2- The system goes to step 2. | |
| **POST CONDITIONS:** | None | |
| **ASSUMPTIONS:** | None | |

|  |  |  |
| --- | --- | --- |
| **USE CASE NAME:** | Remove Research Project | **USE CASE TYPE** |
| **USE CASE ID:** | 28 | **Design Requirements: 🗹** |
| **PRIORITY:** | High |  |
| **PRIMARY BUSINESS ACTOR:** | Research Administrator | |
| **OTHER PARTICIPATING ACTORS:** | None | |
| **DESCRIPTION:** | This use case enables the research administrator to remove research from selected doctor. | |
| **PRE-CONDITIONS:** | The research administrator has logged onto the system. | |
| **TYPICAL COURSE** | Step 1- The research administrator selects the “Remove Research Project” function.  Step 2- The system displays the “Remove Research Project” form with a list of all the doctors (doctor id, last name, and first name) who have research projects in a combo box.  Step 3- The research administrator selects the doctor to remove the research project from.  Step 4- The system displays the doctor’s details (doctor id, last name, first name, and specialty) in a text box.  Step 5- The system displays the research projects’ details (outcome, budget, and research topic description) for each research project linked to the selected doctor in a list box.  Step 6- The research administrator selects the research project to remove.  Step 7- The research administrator clicks on the “remove research” button.  Step 8- The system deletes the research project’s details.  Step 9- The system displays the message “Research project removed successfully”.  Step 10- The system displays the “Remove another research project?” prompt.  Step 11- The research administrator clicks on the “Return” button.  Step 12- The system ends the use case. | |
| **OF EVENTS:** |
|  |
|  |
|  |
|  |
| **ALTERNATE COURSES:** | Step 3A1- The research administrator clicks on the “Return” button.  Step 3A2- The system closes the form. | |
| Step 7A1- The research administrator clicks on the “Return” button.  Step 7A2- The system closes the form. | |
|  | Step 13A1- The research administrator elects to remove another research project.  Step 13A2- The system goes to step 2. | |
| **POST CONDITIONS:** | None | |
| **ASSUMPTIONS:** | None | |

|  |  |  |
| --- | --- | --- |
| **USE CASE NAME:** | Produce Research Projects Report | **USE CASE TYPE** |
| **USE CASE ID:** | 32 | **Design Requirements: 🗹** |
| **PRIORITY:** | High |  |
| **PRIMARY BUSINESS ACTOR:** | Research Administrator | |
| **OTHER PARTICIPATING ACTORS:** | None | |
| **DESCRIPTION:** | This use case enables the research administrator to produce the research projects report. | |
| **PRE-CONDITIONS:** | The research administrator has logged onto the system. | |
| **TYPICAL COURSE** | Step 1- The research administrator selects the “Produce Research Projects Report” function.  Step 2- The system displays the “Research Projects Report” form.  Step 3- The research administrator clicks the “Generate report” button.  Step 4- The system gets the details (research project id, outcome, budget, end date, doctor’s id, last name, first name, and research topic description) of each research project.  Step 5- The system then displays the research project report (research project id, outcome, budget, end date, doctor’s id, last name, first name, and research topic description of each research project.) sorted by research project id.  Step 6- The research administrator clicks on the “Return” button.  Step 7- The system closes the form to end the use case. | |
| **OF EVENTS:** |
|  |
|  |
|  |
|  |
| **ALTERNATE COURSES:** | Step 3A1- The research administrator clicks on the “Return” button.  Step 3A2- The system closes the form. | |
| **POST CONDITIONS:** | None | |
| **ASSUMPTIONS:** | None | |

## 

## 8.5 Sequence Diagrams

### Add Admission

Diagram, schematic

Description automatically generated

### Add Research Project

### Diagram Description automatically generated

### Delete Admission

Diagram

Description automatically generated with medium confidence

### Produce Admission Report

Diagram

Description automatically generated

### Produce Research Projects Report

Graphical user interface, text, application, letter

Description automatically generated

### Remove Prescription

Diagram

Description automatically generated

### Remove Research Project

### Diagram, schematic Description automatically generated

### Update AdmissionDiagram Description automatically generated

## 

## 8.6 Deployment Diagram

Diagram

Description automatically generated

## 8.7 Database Design

Diagram

Description automatically generated

**Data Dictionary**

**Patients**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Required** | **Data Type** | **Maximum Length** | **Range/List** |
| PatientID | Yes | Auto-number | 8 | Primary Key  1-99999999 inclusive |
| LastName | Yes | Text | 25 | - |
| FirstName | Yes | Text | 25 | - |
| StreetAddress | Yes | Text | 50 | - |
| Suburb | Yes | Text | 20 | - |
| City | Yes | Text | 20 | - |
| EmailAddress | Yes | Text | 30 | - |
| PhoneNumber | No | Number | 15 | - |
| InsuranceCode | Yes | Text | 7 | - |

**Admissions**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Required** | **Data Type** | **Maximum Length** | **Range/List** |
| AdmissionID | Yes | Auto-number | 8 | Primary Key |
| Description | Yes | Text | 30 | - |
| AdmissionDate | Yes | Date | 10 | Format: DD/MM/YYYY |
| Status | Yes | Text | 20 | Current, Complete |
| WardID | Yes | Foreign Key (WARD) | 2 | - |
| PatientID | Yes | Foreign Key (PATIENT) | 3 |  |

**Wards**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Required** | **Data Type** | **Maximum Length** | **Range/List** |
| WardID | Yes | Auto-number | 2 | Primary Key |
| WardName | Yes | Text | 20 | - |
| Location | Yes | Text | 10 | - |
| Capacity | Yes | Number | 2 | 1 to 20 inclusive |

**Payments**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Required** | **Data Type** | **Maximum Length** | **Range/List** |
| PaymentCode | Yes | Auto-number | 8 | Primary Key |
| AdmissionID | Yes | Foreign Key (ADMISSION) | 8 | - |
| Amount | Yes | Currency | 6 | - |
| Date | Yes | Date | 10 | Format: DD/MM/YYYY |

**Doctors**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Required** | **Data Type** | **Maximum Length** | **Range/List** |
| DoctorID | Yes | Auto-number | 4 | Primary Key |
| LastName | Yes | Text | 25 | - |
| FirstName | Yes | Text | 25 | - |
| StreetAddress | Yes | Text | 50 | - |
| Suburb | Yes | Text | 20 | - |
| City | Yes | Text | 20 | - |
| Specialty | Yes | Text | 30 | - |
| PhoneNumber | No | Text | 20 | - |
| Salary | Yes | Number | 6 | 20000.00-200000.00 inclusive |

**Prescriptions**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Required** | **Data Type** | **Max Length** | **Range/List** |
| PrescriptionID | Yes | Auto-Number | 8 | Primary Key |
| MedicationName | Yes | Foreign Key (ADMISSION) | 30 | - |
| PrescriptionDate | Yes | Date | 10 | Format: DD/MM/YYYY |
| Amount | Yes | Number | 3 | 1 to 150 inclusive |
| AdmissionID | Yes | Foreign Key (ADMISSION) | 8 | - |

**Medication**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Required** | **Data Type** | **Max Length** | **Range/List** |
| MedicationID | Yes | Auto-number | 6 | Primary Key |
| MedicationName | Yes | Text | 30 | - |
| Cost | Yes | Currency | 6 | 0.50 to 9999.99 inclusive |

**Research Project**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Required** | **Data Type** | **Maximum Length** | **Range/List** |
| ResearchProjectID | Yes | Auto-number | 6 | Primary Key |
| Outcome | Yes | Text | 30 | Journal article, conference paper, conference poster, book chapter, and book |
| EndDate | Yes | Date | 10 | Format: DD/MM/YYYY |
| Budget | Yes | Currency | 7 | - |
| ResearchTopicID | Yes | Foreign Key (RESEARCHTOPICS) | 3 | - |
| DoctorID | Yes | Foreign Key (DOCTORS) | 4 | - |

**Research Topic**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Required** | **Data Type** | **Maximum Length** | **Range/List** |
| ResearchTopicID | Yes | Auto-Number | 3 | Primary Key |
| Description | Yes | Text | 30 | - |
| Level | Yes | Number | 5 | - |

**Allocation**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Required** | **Data Type** | **Maximum Length** | **Range/List** |
| AllocationID | Yes | Auto-number | 6 | Primary Key |
| JobFee | Yes | Currency | 5 | Between 1 to 99999 |
| Role | Yes | Text | 30 | - |
| AdmissionID | Yes | Foreign Key (ADMISSION) | 3 | - |
| DoctorID | Yes | Foreign Key (DOCTORS) | 4 | - |

## 8.8 Annotated User Interface Designs

*Diagram

Description automatically generatedChart

Description automatically generated with low confidenceDiagram

Description automatically generatedDiagram

Description automatically generatedGraphical user interface, diagram

Description automatically generatedDiagram

Description automatically generatedDiagram, schematic

Description automatically generatedDiagram

Description automatically generated*

## 8.9 Test Plan

# 9. Project Training

## 9.1 End User Background and Training Objectives

## 9.2 Training Materials

## 9.3 Training Deliverables

## 9.4 Evaluation

# 10. Conclusion & Lessons Learned

# References

[Five Advantages of Agile Software Development Methodologies | EagleDream Technologies](https://eagledream.com/news/application-modernization-news/5-advantages-agile-software-development-methodologies/)

[The Pros and Cons of Waterfall Methodology | Lucidchart Blog](https://www.lucidchart.com/blog/pros-and-cons-of-waterfall-methodology)

[What is WaterFall Model in Software Developement Life Cycle | SDLC (toolsqa.com)](https://www.toolsqa.com/software-testing/waterfall-model/)

# Appendices