Institiúid Teicneolaíochta Chorcaí
Cork Institute of Technology

We can confirm the following details:

Student ID/Registration Number: R00117318

R00109672

Name: Adam Lloyd

Jade O'Sullivan

Module Name: OO Analysis and Design

Module Lecturer: Mary Davin

We can confirm that this is our own work

Due Date: Sunday 17th May 2015

Table of Contents

Part One

i) Identify any 4 more use cases

1. Record Treatments Performed

2. Carry out Pre- Admission Assessment

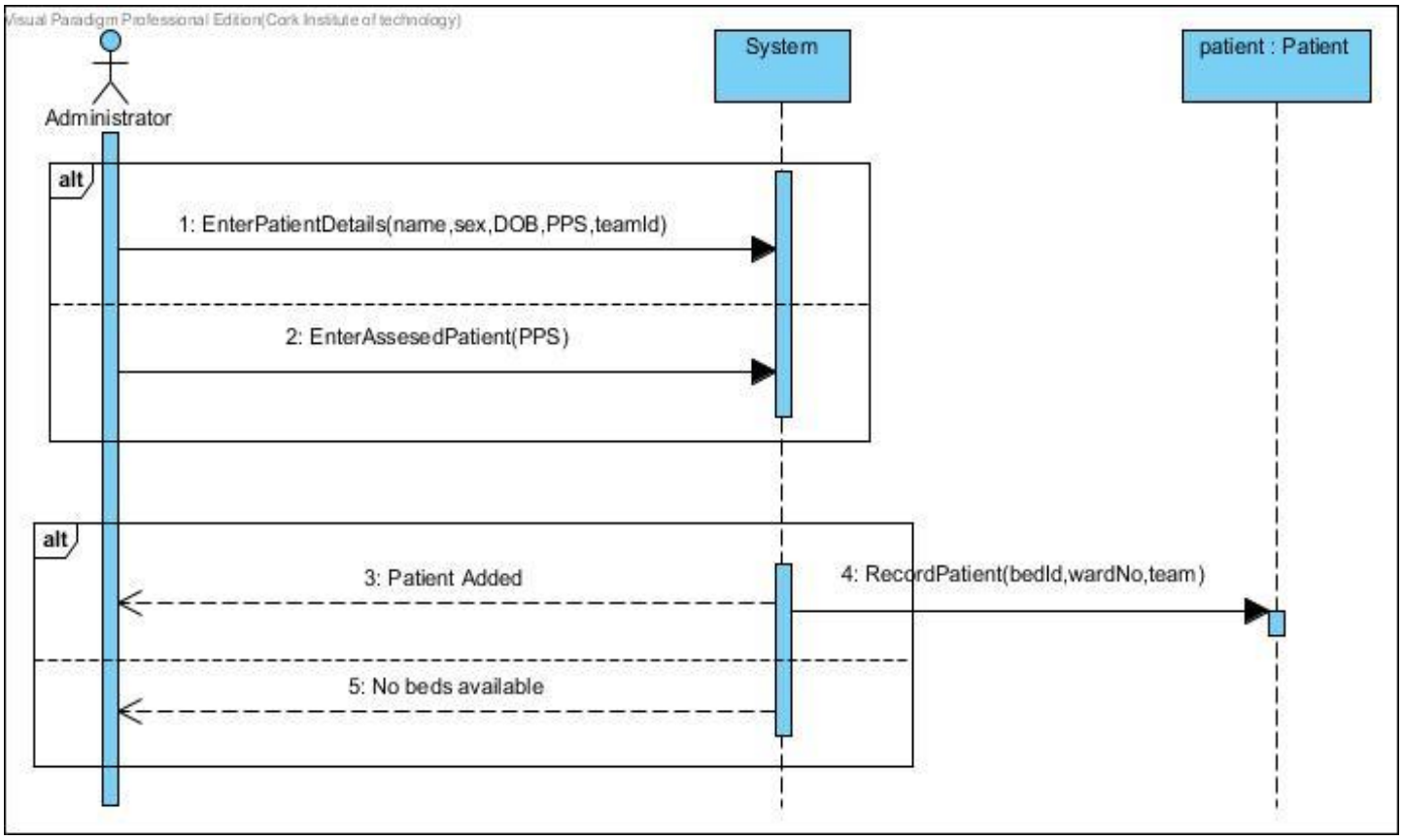
3. Record Pre-Admission Assessment Results

4. Record Patient Treatments

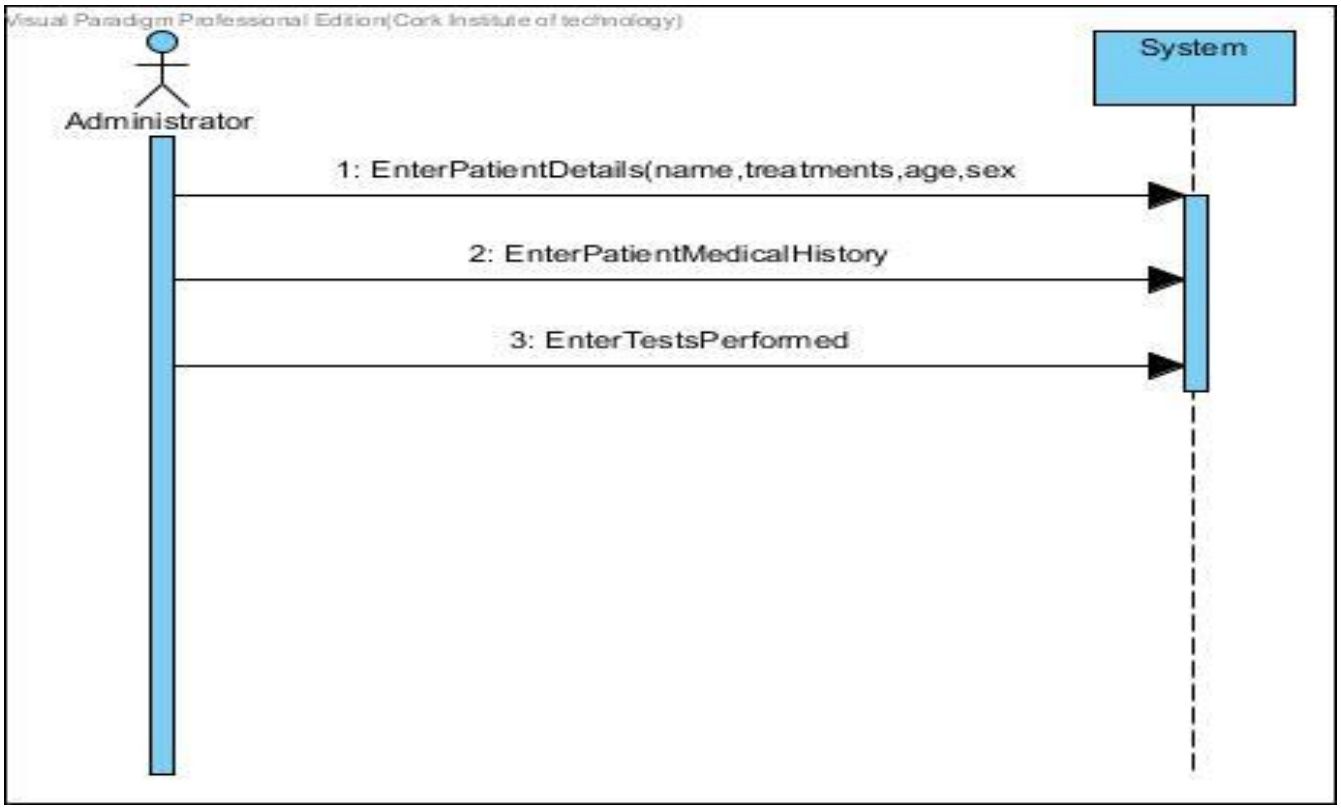
ii) Use Case Diagram

iii) System Sequence Diagrams

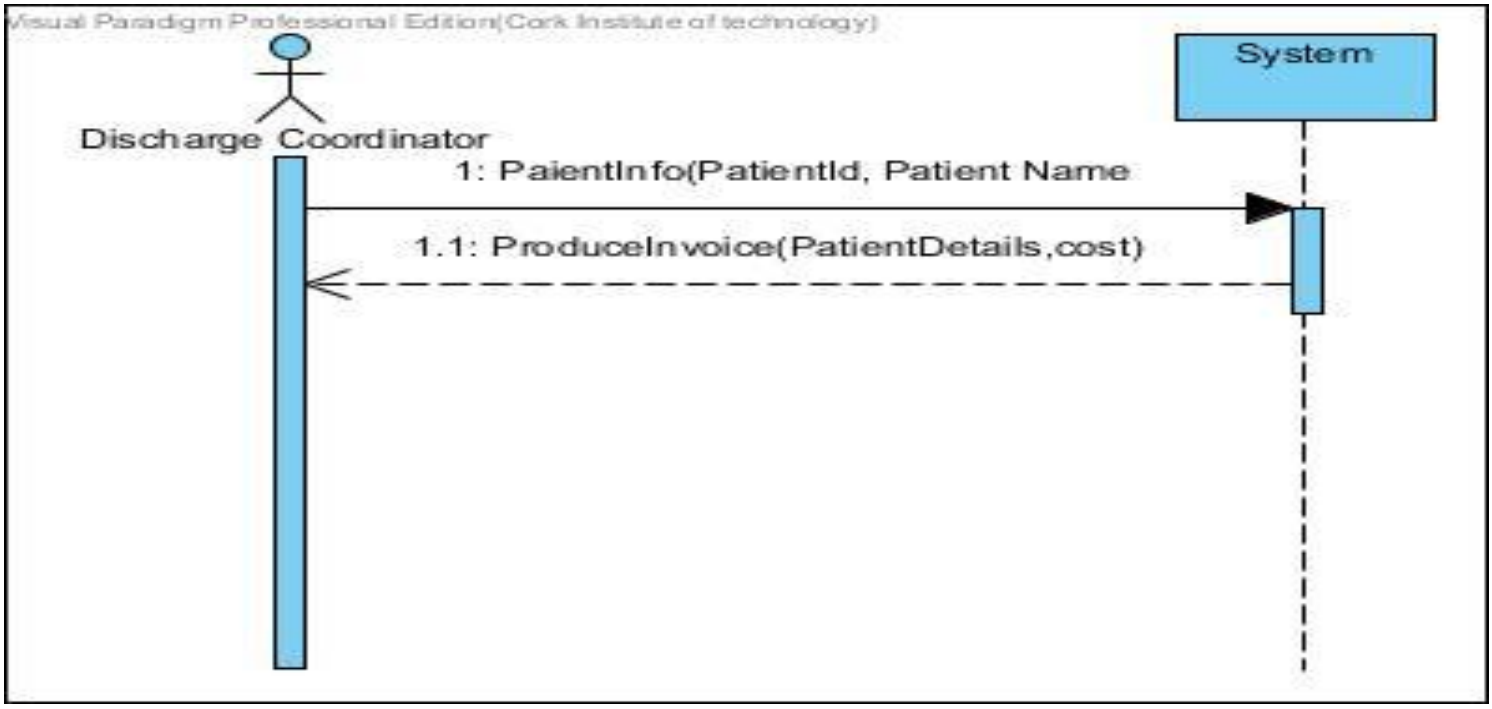
Admit Patient



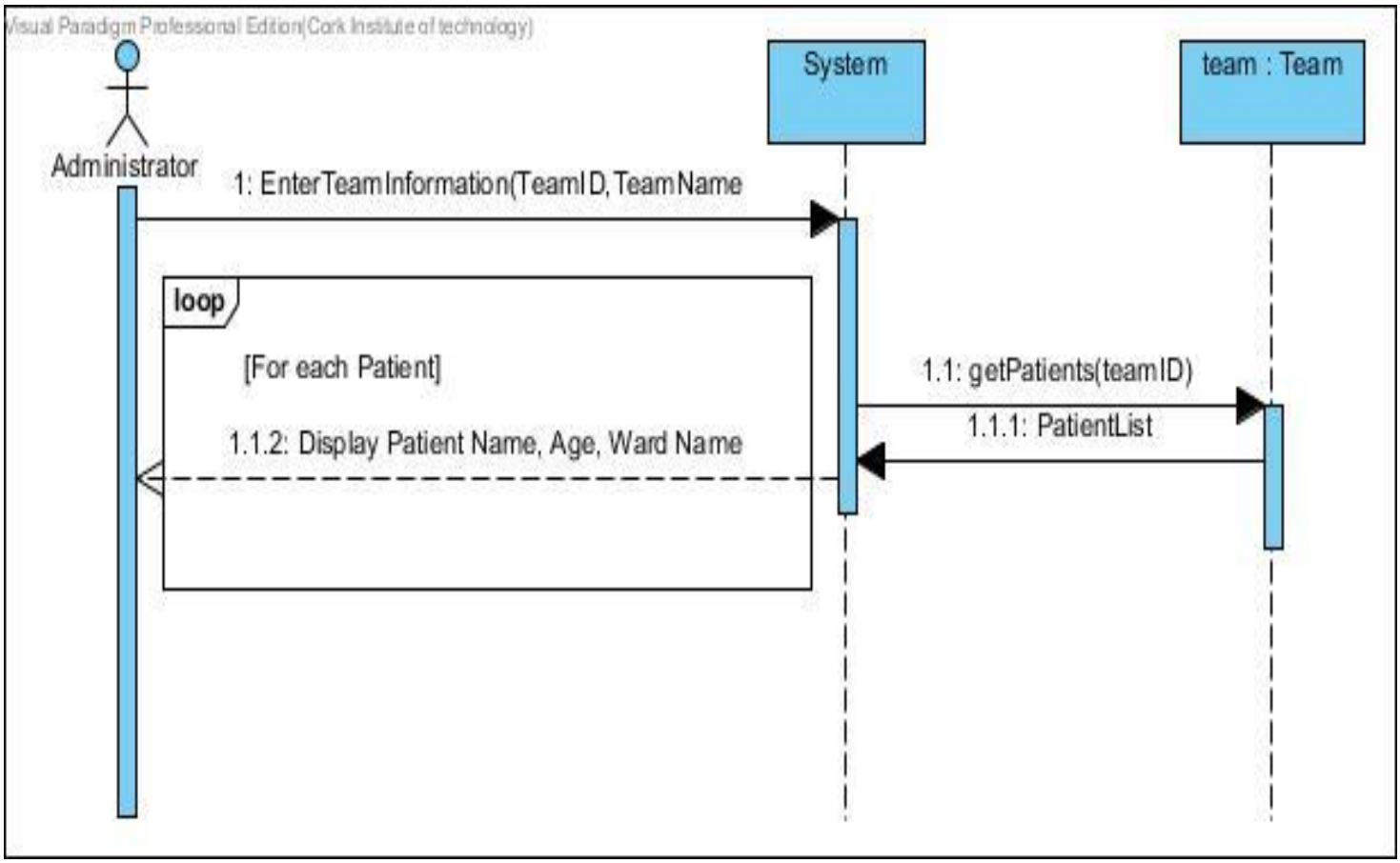
Carry Out Pre Admission Assessment



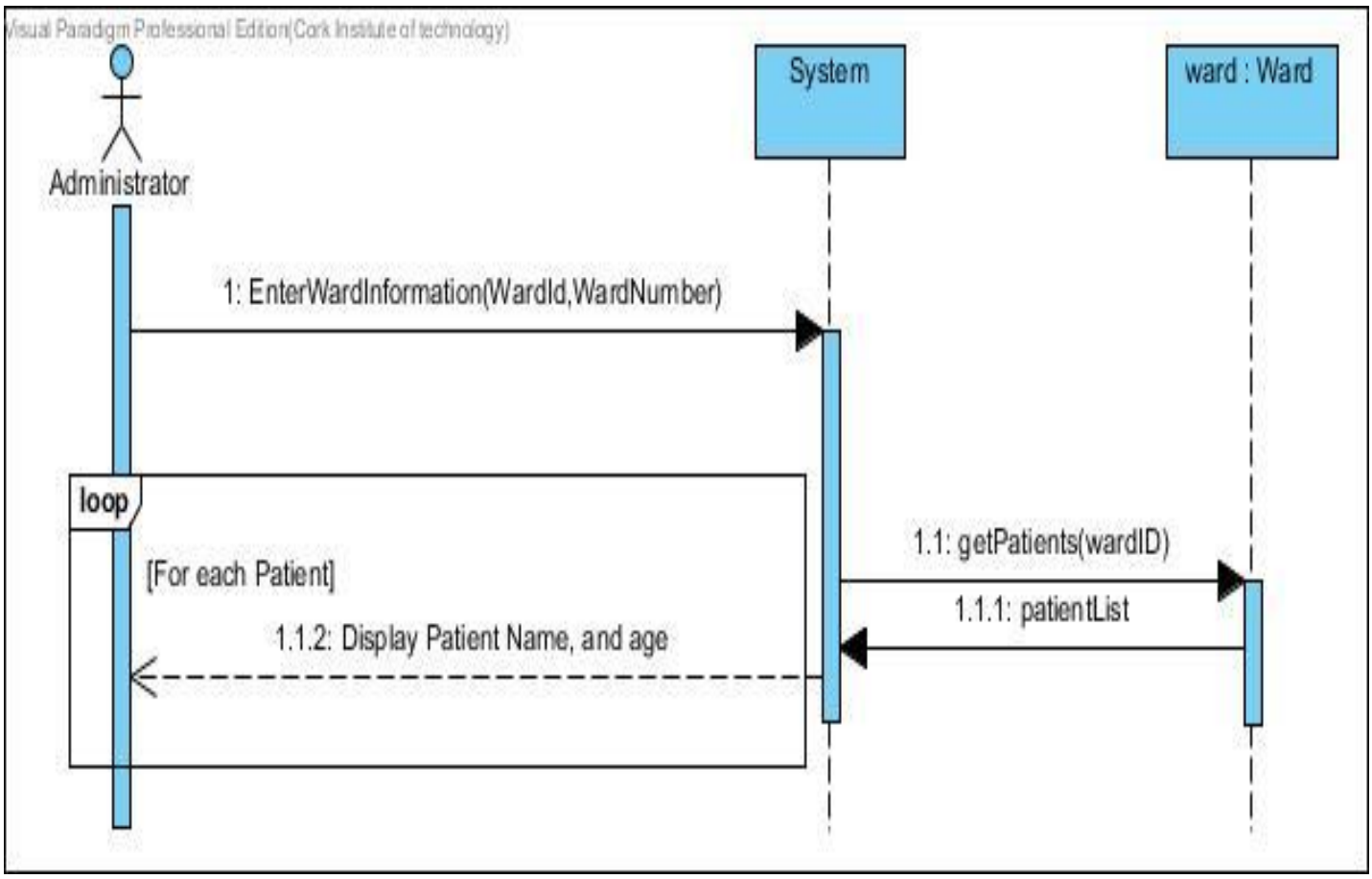
Discharge Patient



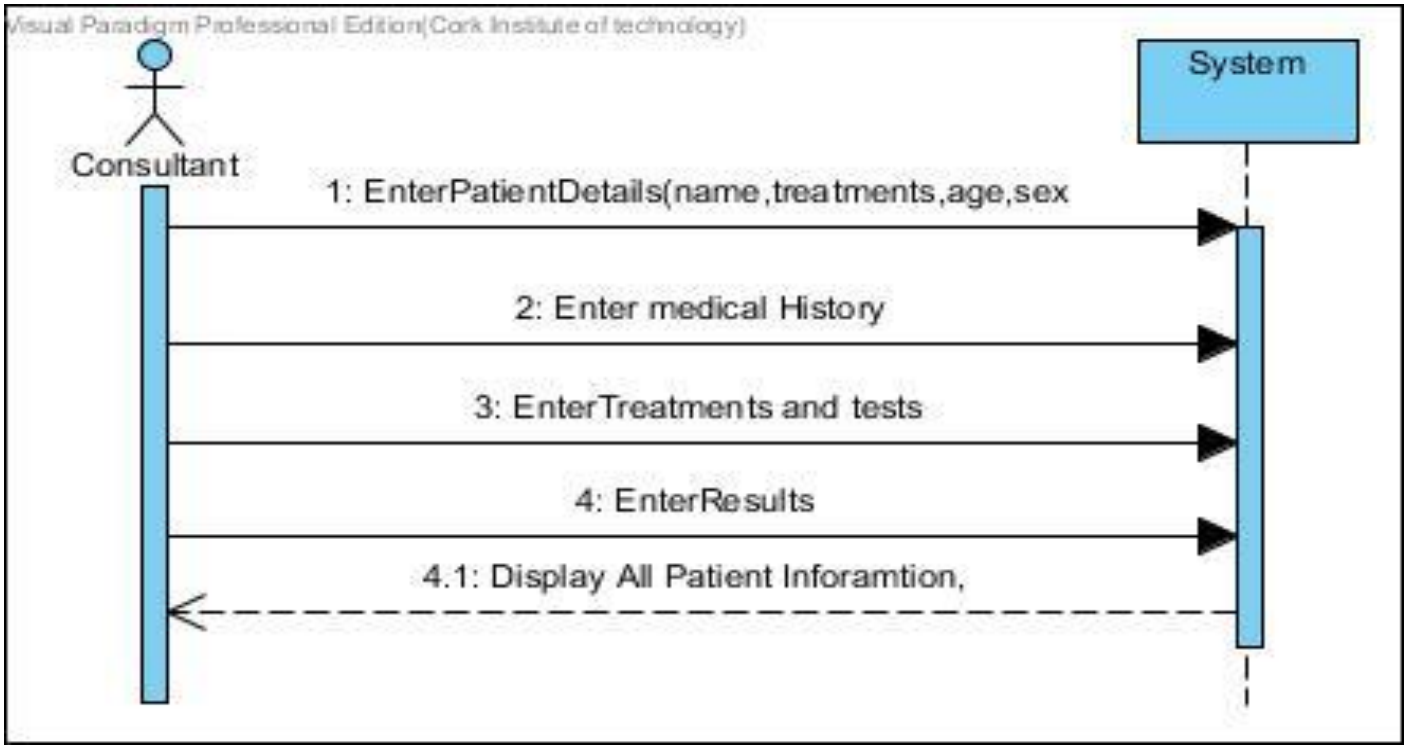
List Patients Treatments



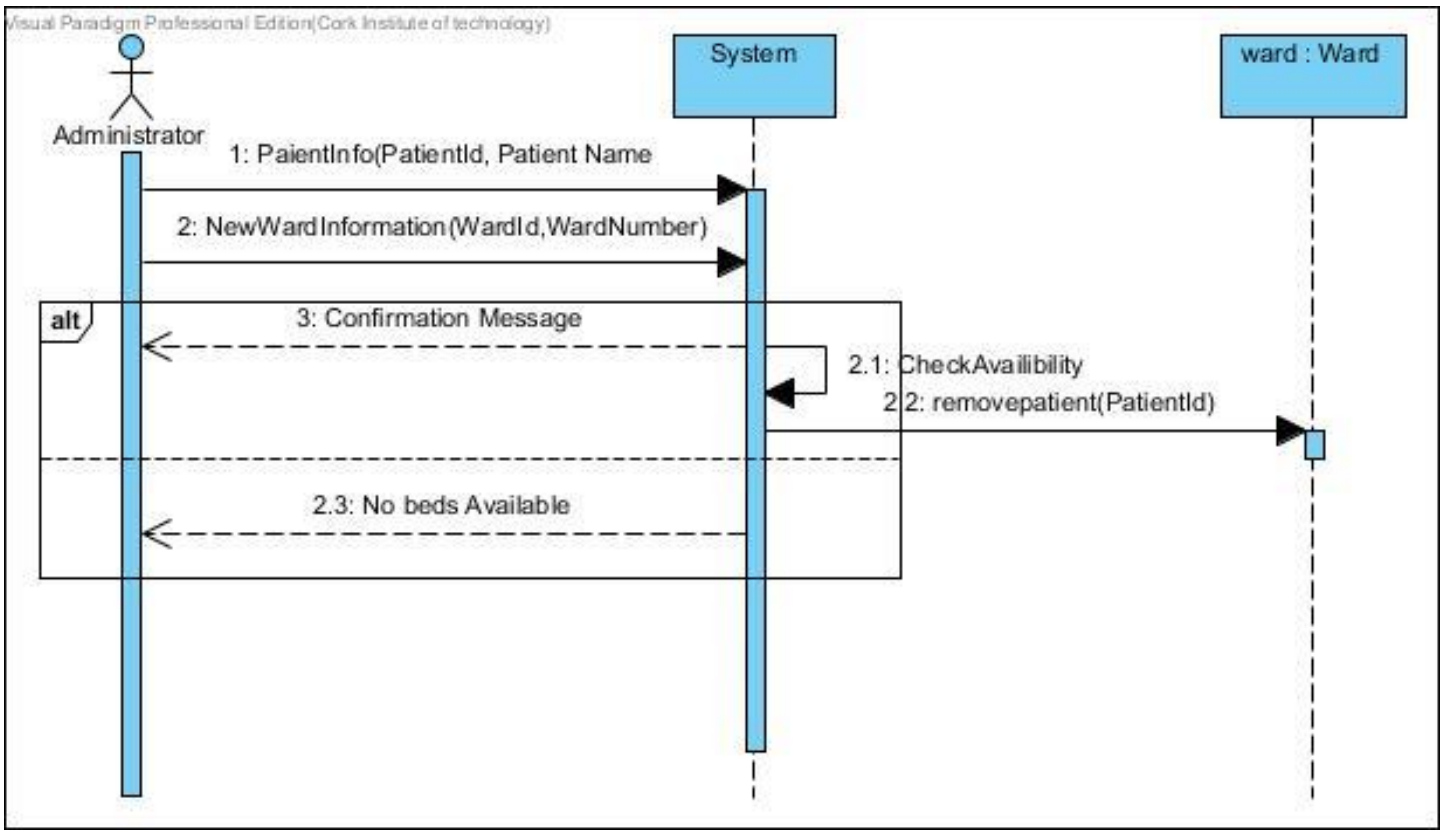
List Teams Patients



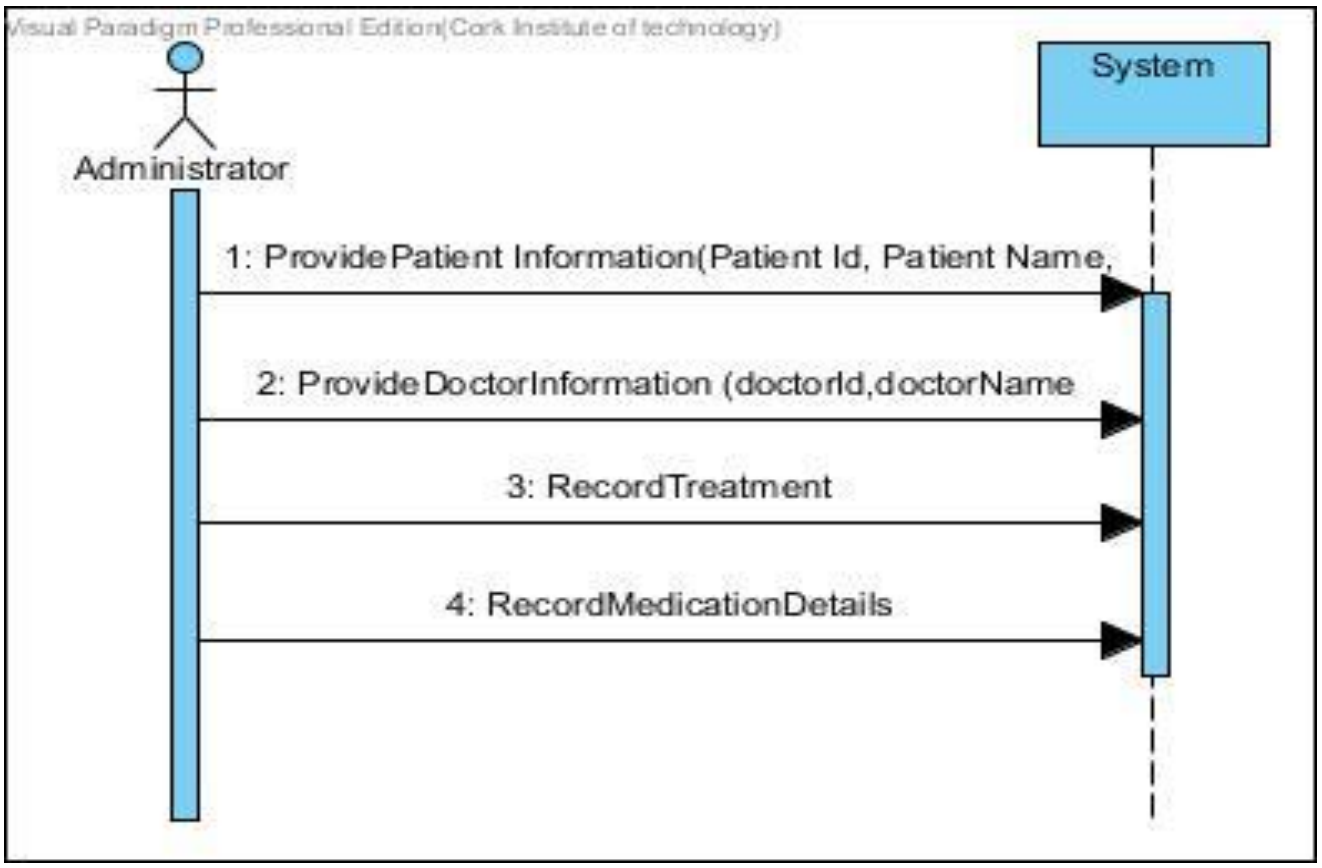
List Wards Patients



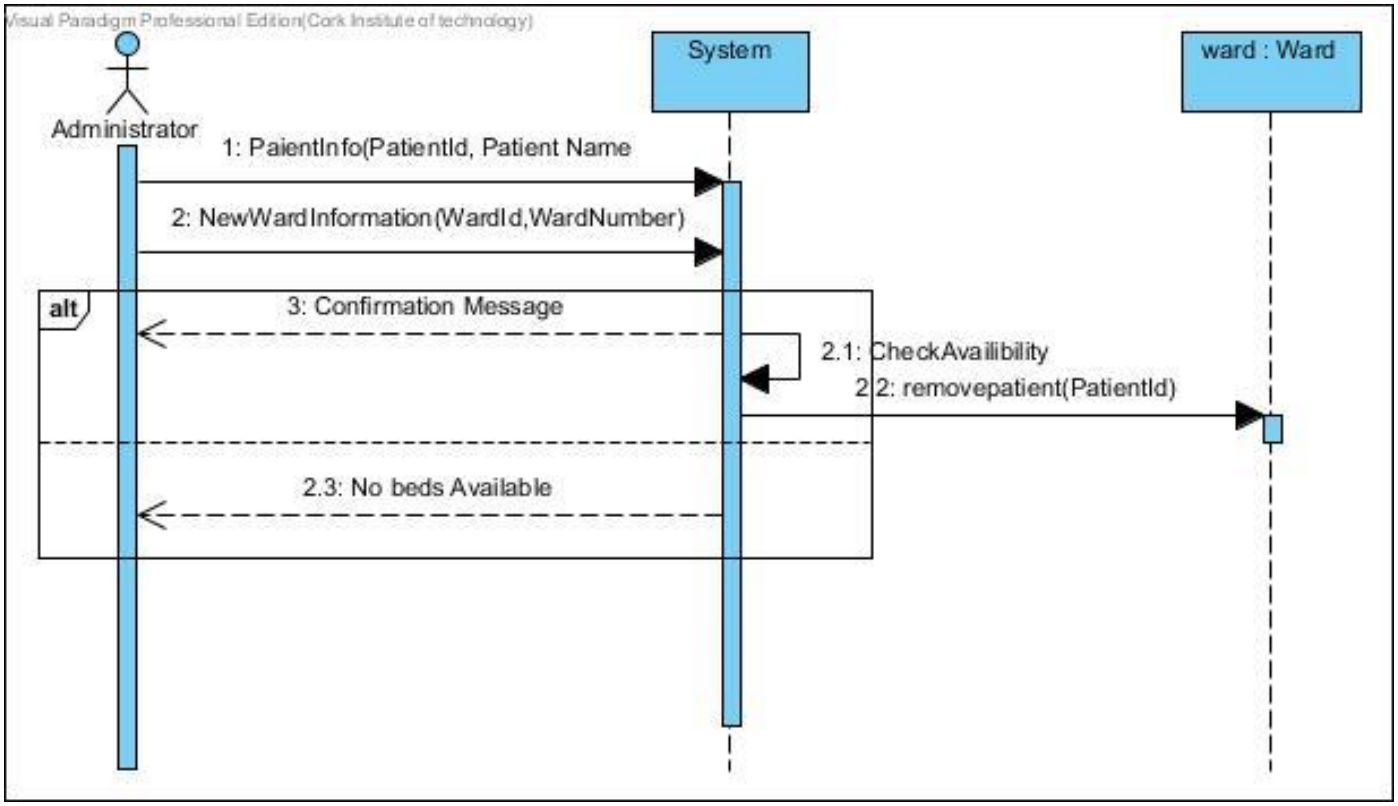
Record Pre Admission Assessment Results



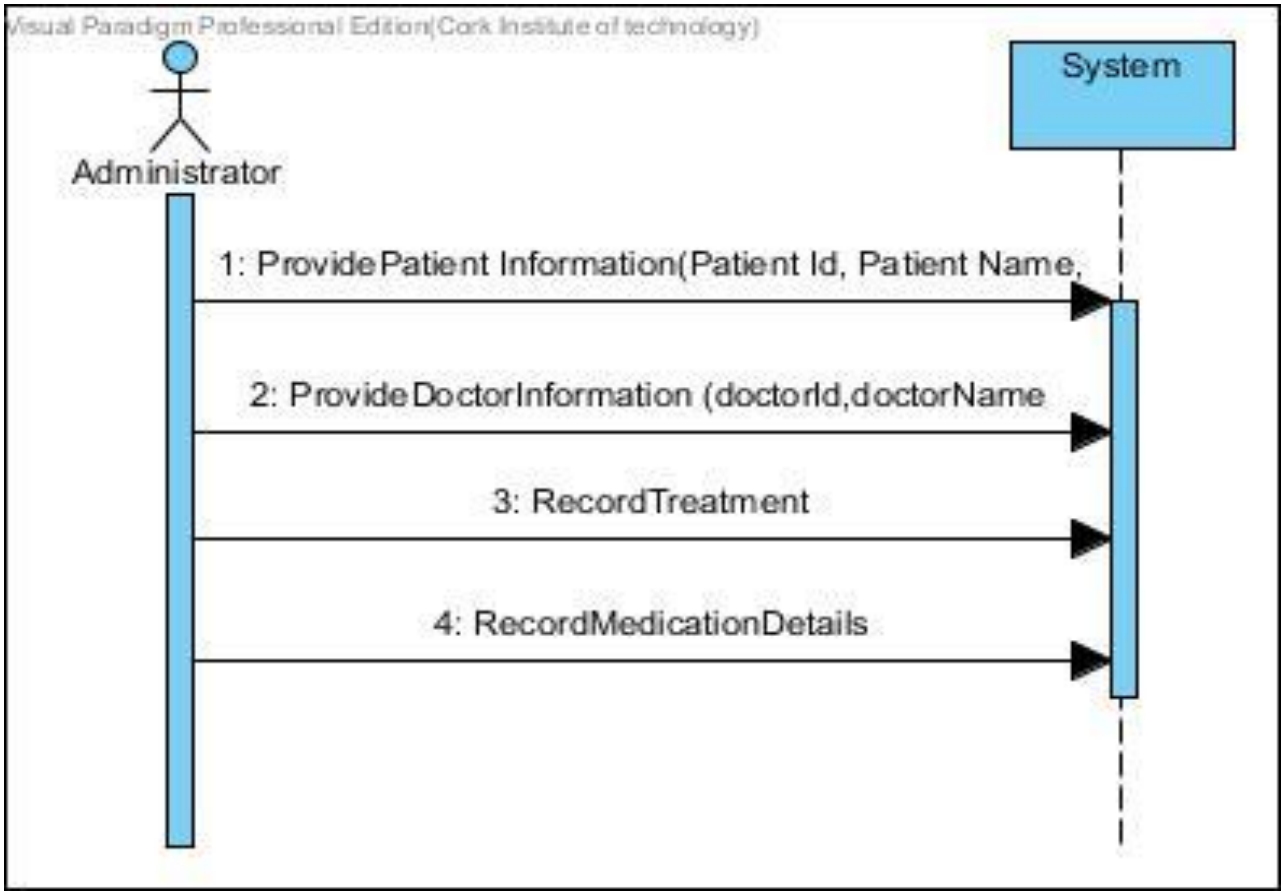
Record Treatments Performed



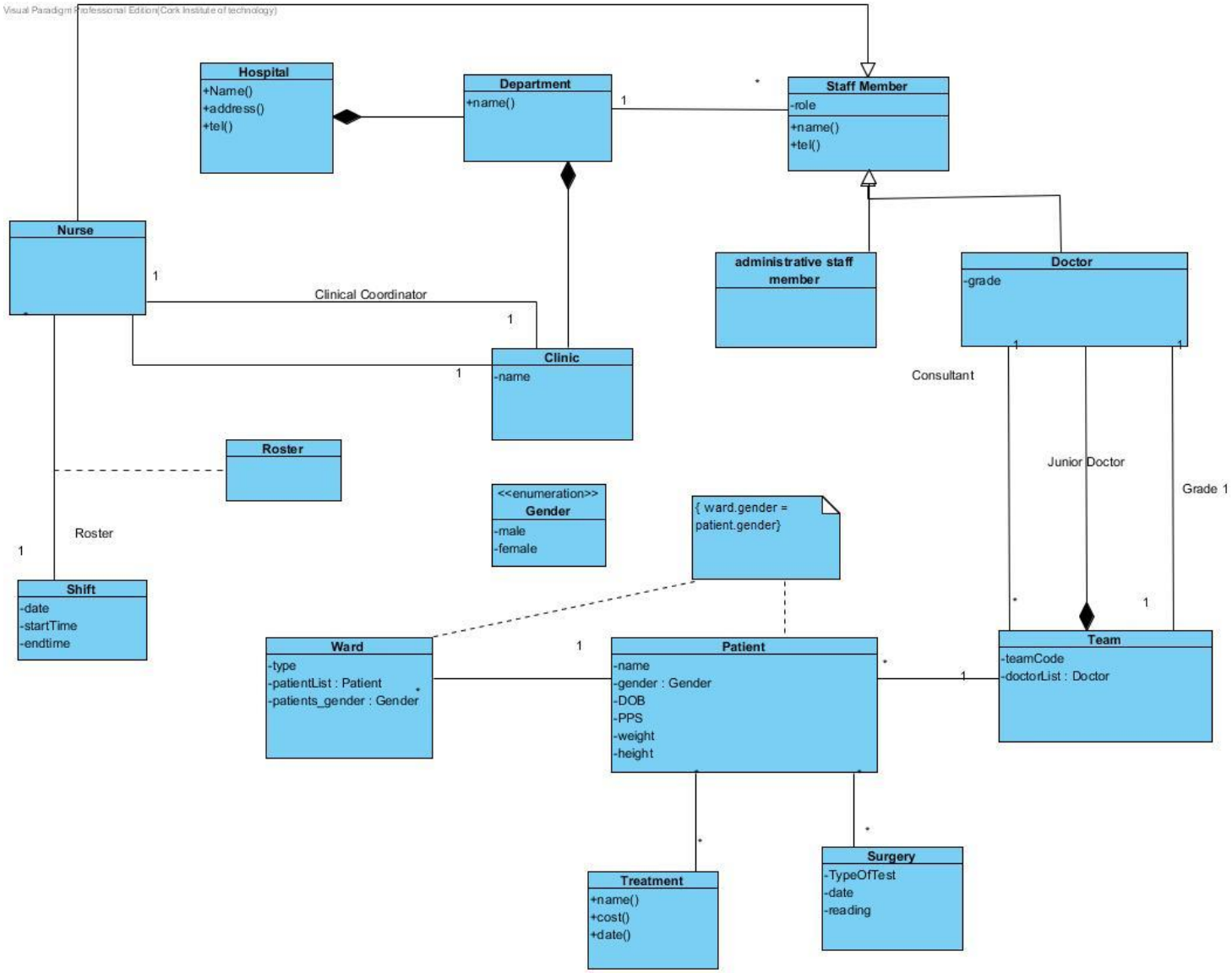
Transfer Patient



Treat Patient

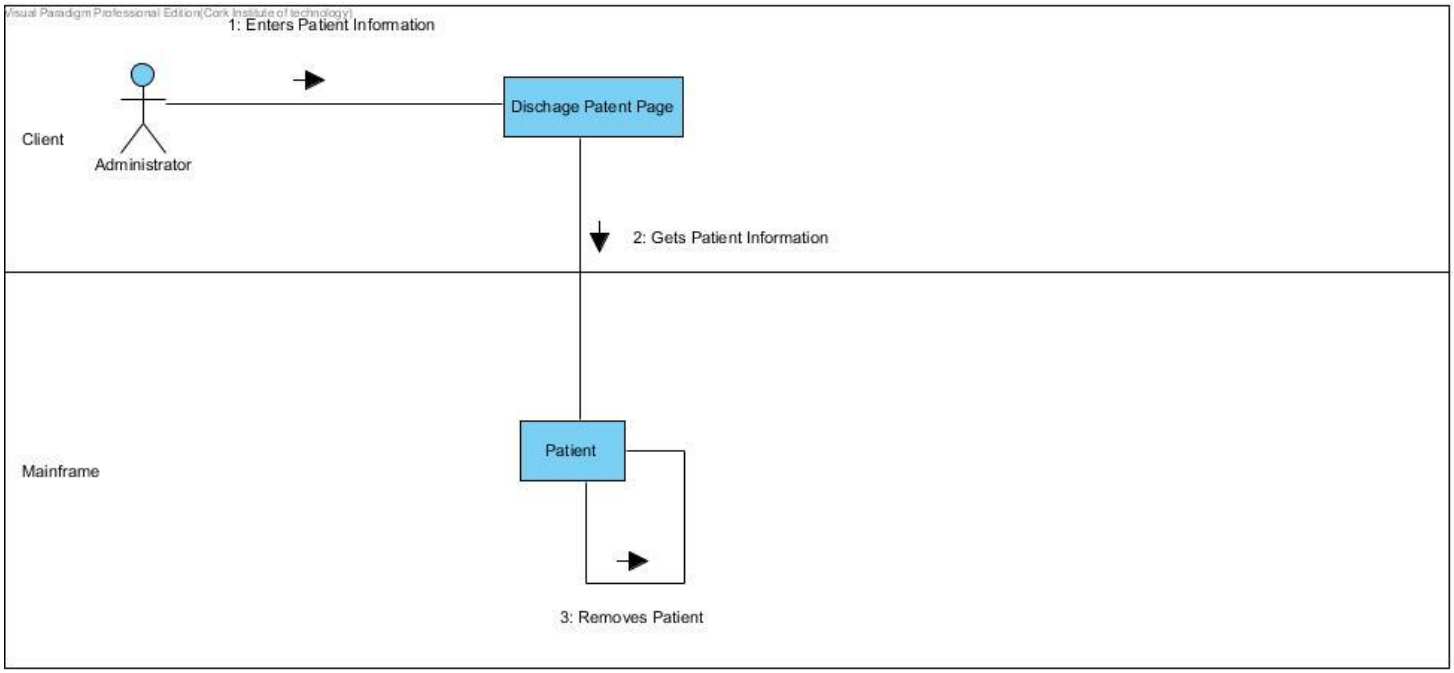


iv) Domain Model

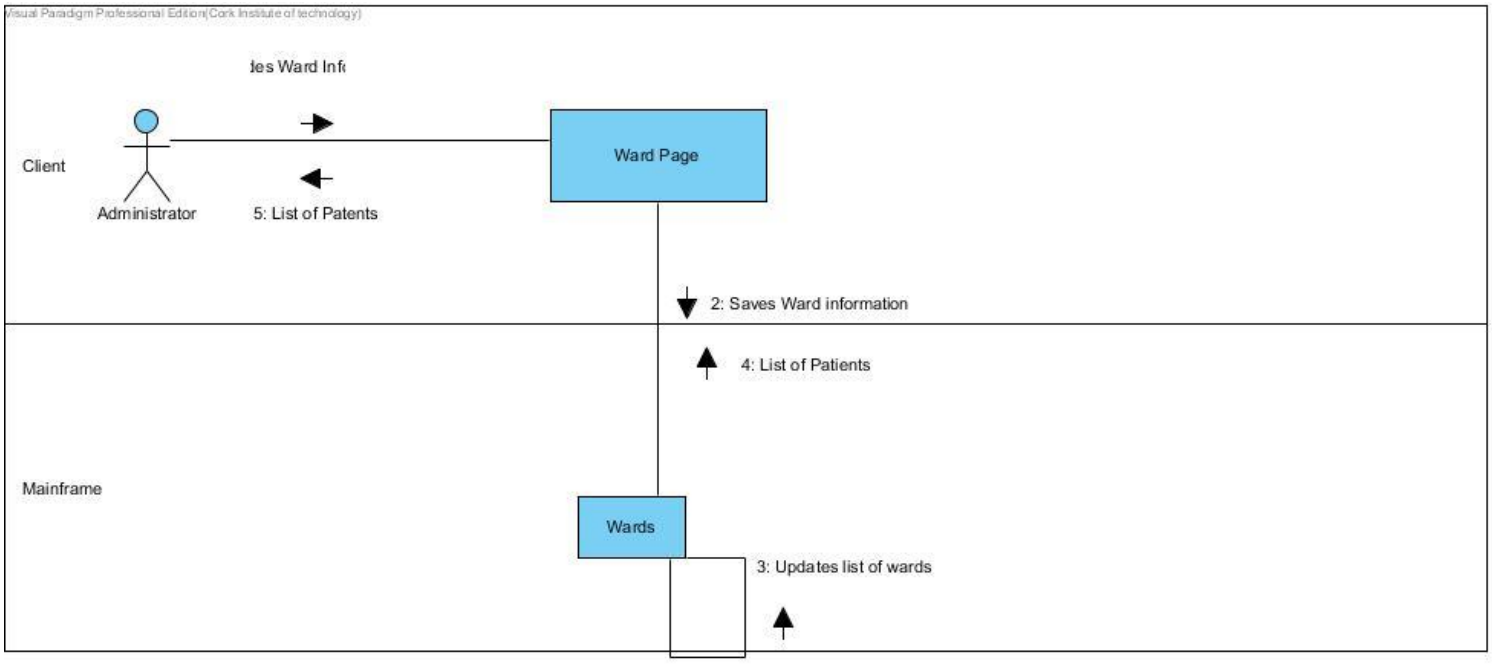


v) Sequence/Communication Diagrams

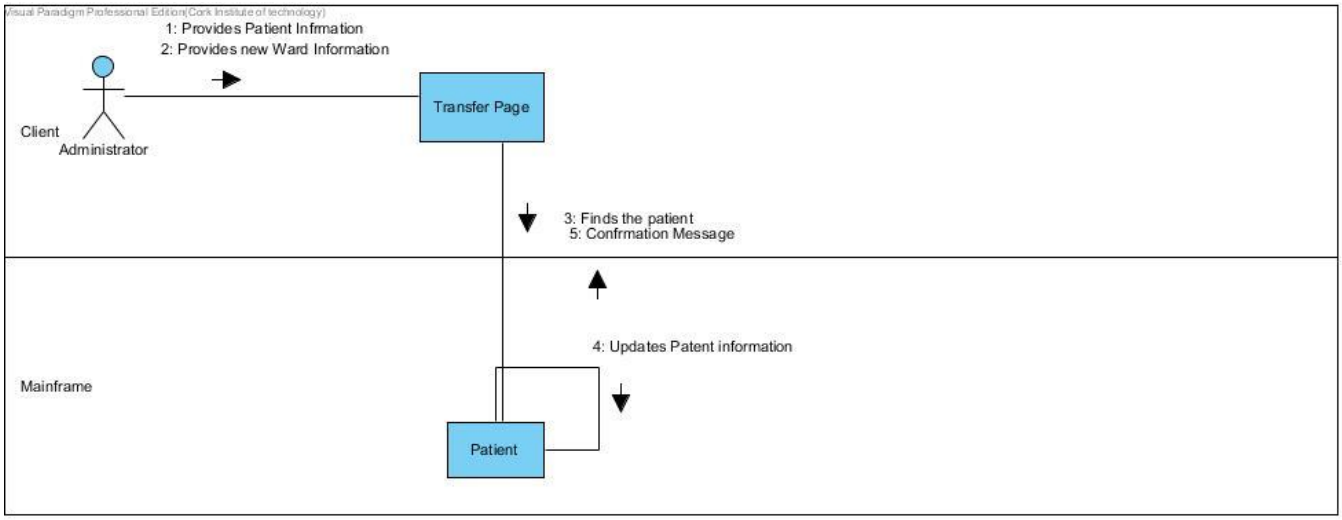
Admit Patient



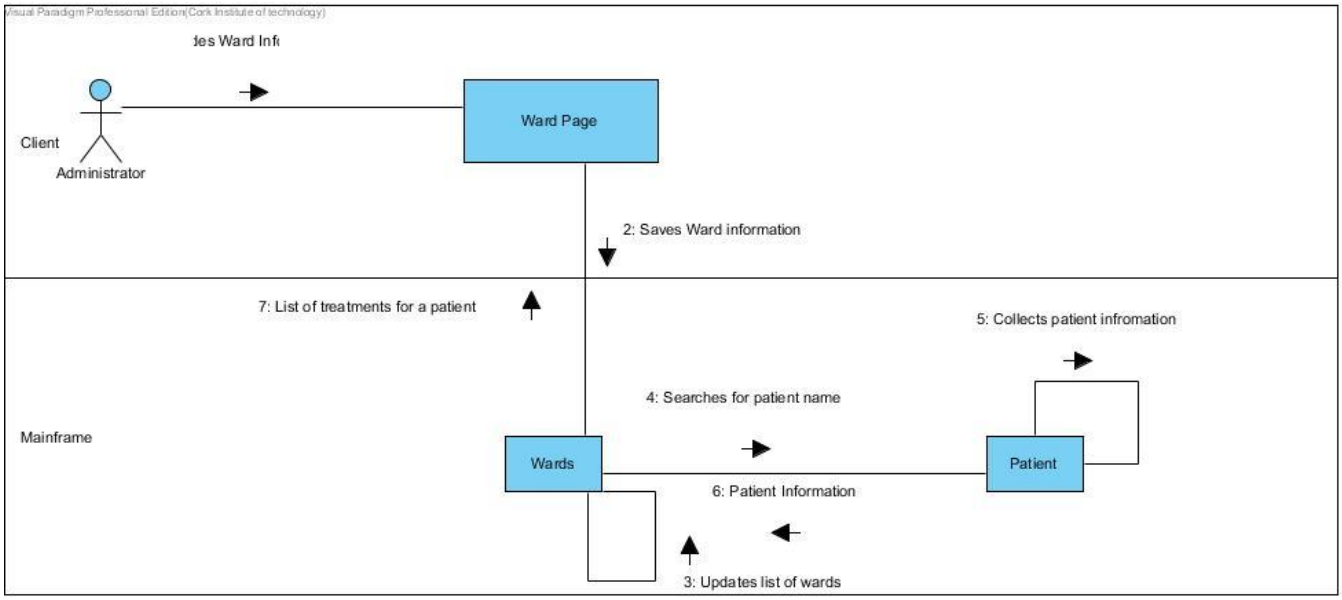
Get Wards Patients



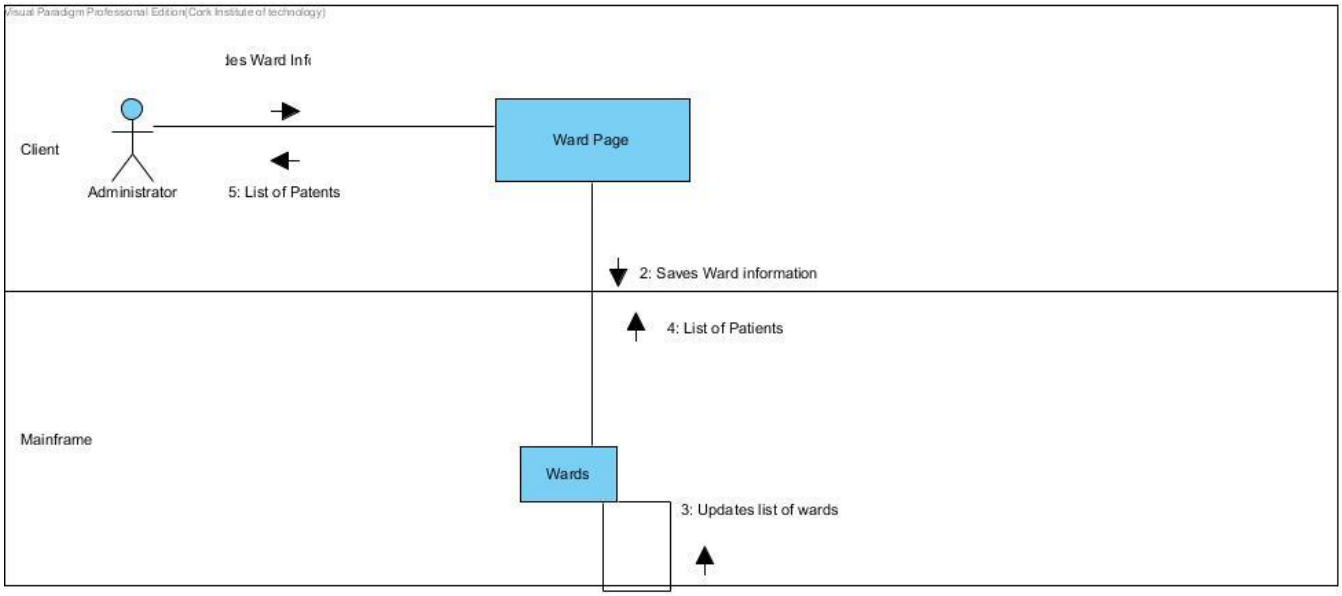
Transfer Patient



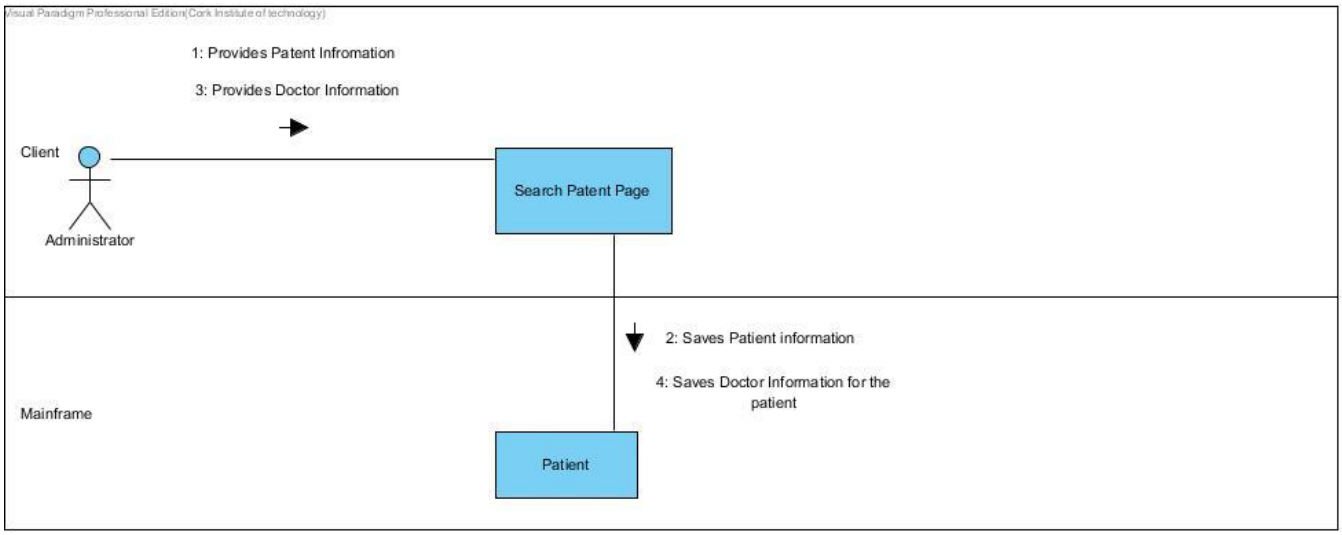
List Patient Treatments



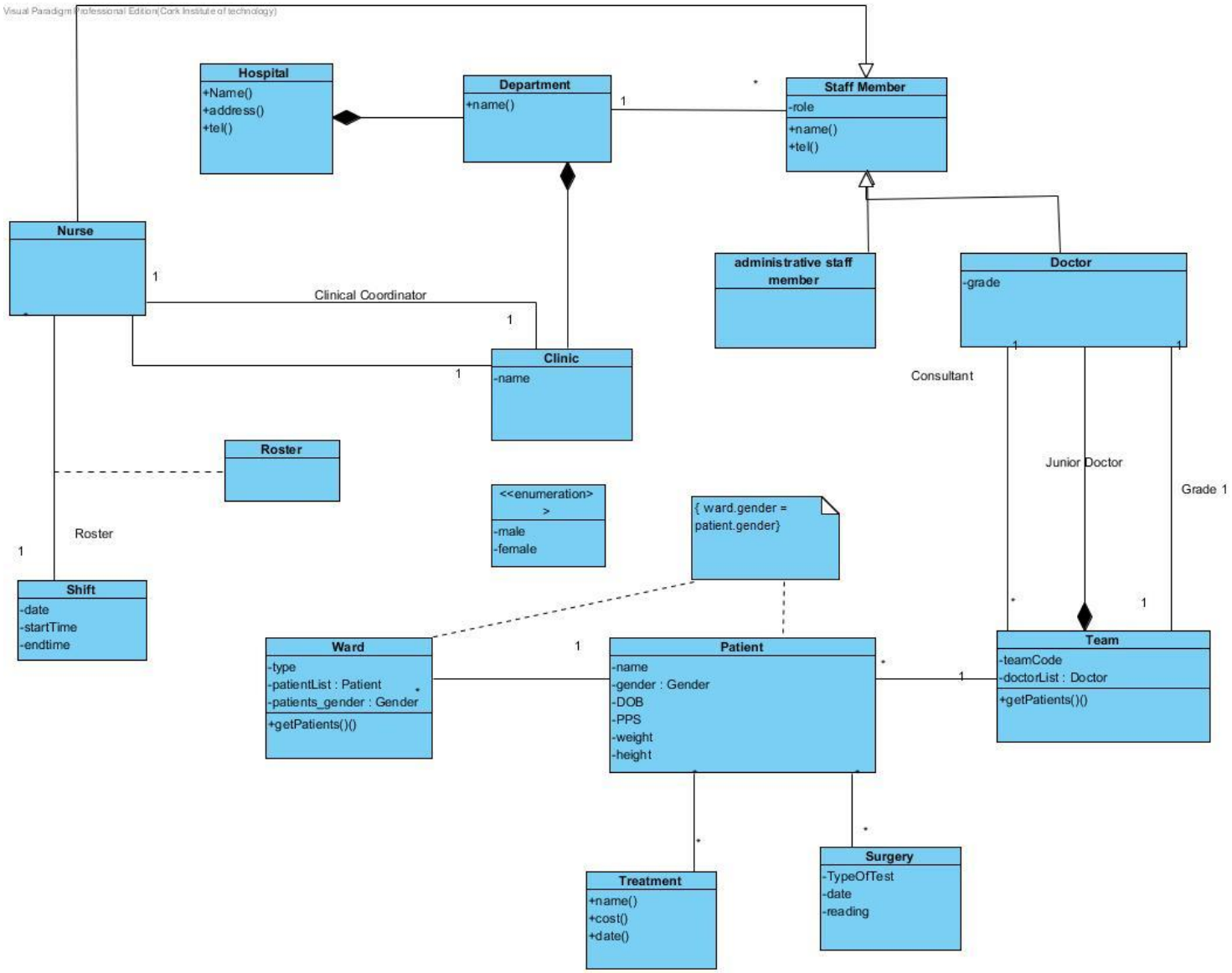
List teams patients



Treat Patient



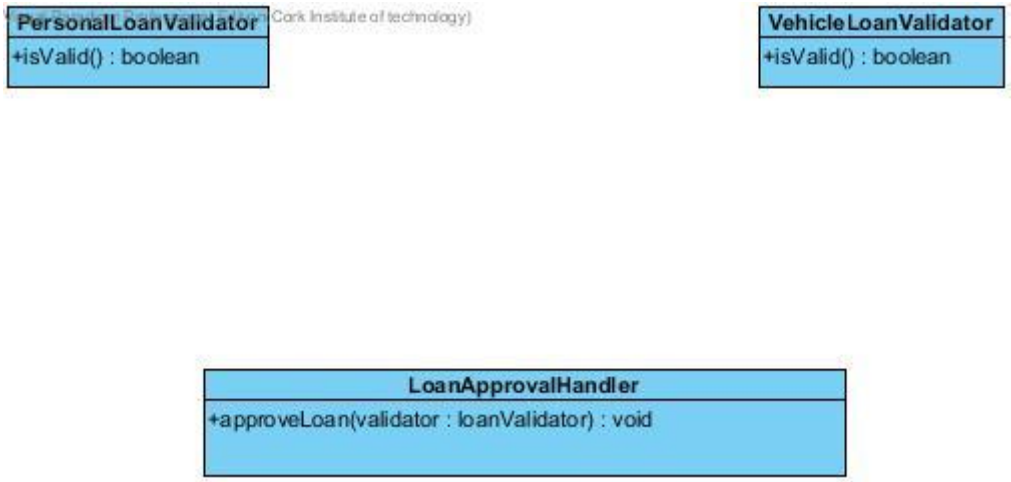
vi) Design Class Diagram



Part Two

Q3:

i)Class Diagram to Model the code



ii) Open Closed Principle:

The Open Closed Principle states that a 'module should be open for extension and closed for modification.' This means an entity's behaviour can be extended without modifying its source code.

In this program, the code needs to be modified for every type of loan we have. We can have different types of loans from person to Vehicle and different types of validators also.

iii)Rewritten Code

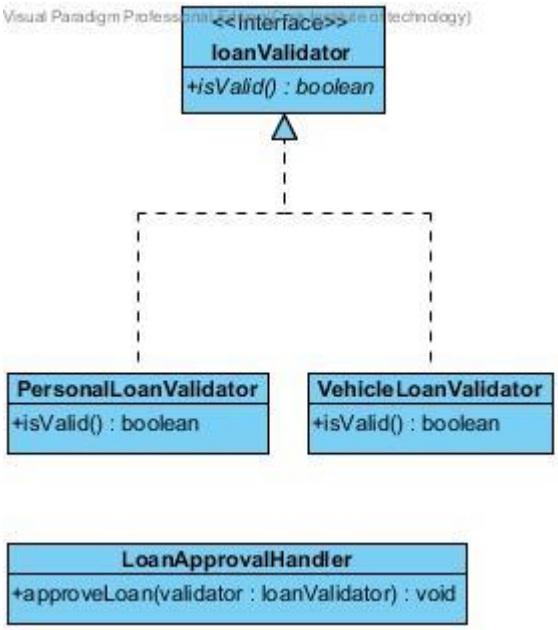
```
//Interface
public interface loanValidator {
      public boolean isValid();
}


public class VehicleLoanValidator implements loanValidator
{
  public boolean isValid()
  {
    return false;
    //Validation logic
  }
}

public class PersonalLoanValidator implements loanValidator
{
  public boolean isValid()
  {
    return false;
    //Validation logic
  }
}

public class LoanApprovalHandler
{
  public void approveLoan (loanValidator validator)
  {
    if ( validator.isValid())
    {
      //Process the loan.
    }
  }
}
```

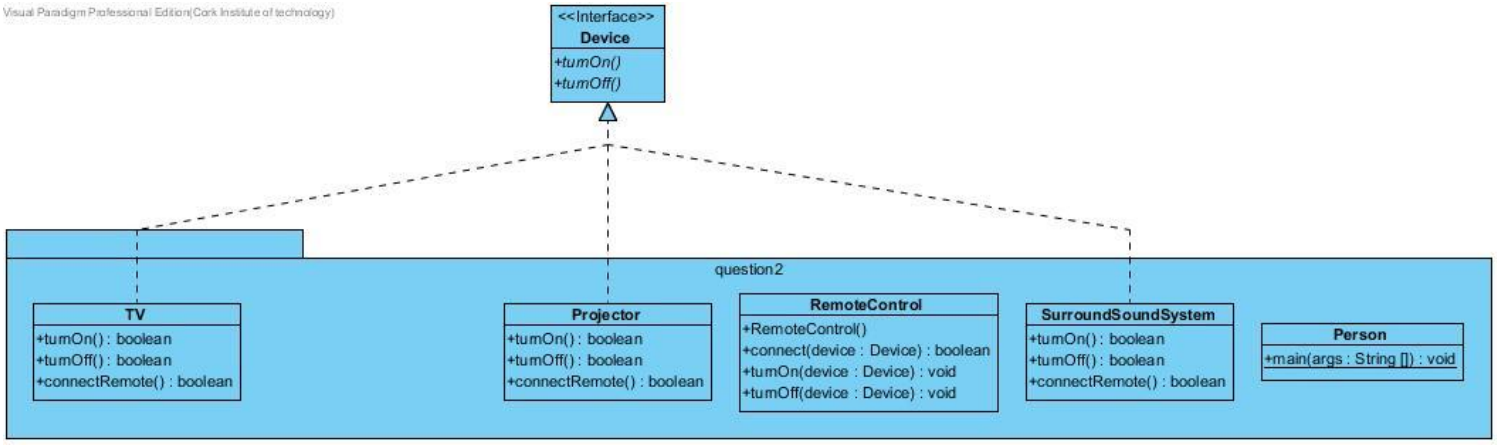
iv) Corrected Code Class Diagram



Q2: Design Class Diagram to Model Home Entertainment System

Question was unclear if it was required to write and reverse code or not so did it both ways.

Reversed from code:



Modelled without code