|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Qualification details** | | | | | |
| **Training Package Code and Title** | ICT - Information and Communications Technology (Release 8.0) | | | | |
| **Qualification National**  **Code & Title:** | ICT40120 Certificate IV in Information Technology (Release 4) | | | **State code** | BFF9 |
| **Qualification National**  **Code & Title:** | ICT50220 Diploma of information Technology (Release 2) | | | **State code** | BGJ4 |
| **Assessment Title** | Assessment Task Three (Individual Project) | | | | |
| **Unit National Code & Title** | ICTPRG443 Apply intermediate programming skills in different languages | | | | |
| ICTICT430 Apply software development methodologies | | | | |
| ICTICT449 Use version control systems in development environments | | | | |
| **Date Due** | Week Nineteen | | **Date Received** |  | |
| **Student Name** | Joshua Farrell | | | | |
| **Student Declaration** | I declare that the evidence submitted is my own work: | | | | |
| **Assessor Name** |  | | | | |
| **Assessment Decision** | Satisfactory | | Not Yet Satisfactory | | |
| **Assessor Signature** |  | | **Date** |  | |
| **Is student eligible for reassessment (Re-sit)?** | No | Yes | **Reassessment Date:** | Week Twenty | |
| **Feedback to student** | | | | | |
| *Via Blackboard (LMS) – Please check [Grade] section.* | | | | | |
| **Feedback from student** | | | | | |
| *Via Blackboard (LMS) – Please use [Comment] section during submission.* | | | | | |
| **Student signature** |  | | **Date** |  | |

|  |  |
| --- | --- |
| **Assessment Instructions** | |
| **TO THE ASSESSOR** | |
| Type of Assessment | Individual Project |
| Duration of Assessment | 5 Class Sessions (Week 15 - 19) |
| Location of Assessment | Classroom |
| Conditions | Assessor to ensure that the noise levels, natural interactions and time variances are maintained as it would be in the Software Development industry.  Learners are required to complete the required tasks in class and submit the required documentation electronically via Blackboard |
| Elements and Criteria | As detailed in the assessment plan  You are required to make sure that all students meet the elements, performance criteria and oral communication items as outlined in the provided solution. |
| **TO THE STUDENT** | |
| Purpose of Assessment | You are required to show you can:  ICTPRG443 Apply intermediate programming skills in different languages   * Demonstrate your skills and knowledge by creating, coding, debugging and testing code * Establish user requirements and then research and collect information about data structures to provide suitable solutions. * Manage time and tasks to complete a series of coding and documentations problems   ICTICT430 Apply software development methodologies   * Select traditional and non-traditional systems development methodologies * Apply selected software methodology to project plan which identifies resources and control structures * Document analysis for approval to external stakeholders.   ICTICT449 Use version control systems in development environments   * Prepare and evaluate version control systems * Install and configure a version control system * Create and upload code to version control system * Test and review logs on version control system   The student must demonstrate the ability to complete the tasks outlined in this assessment and is expected to use systematic analytical processes and effect time management to meet the goals/deadlines outlined in the DAP. |
| Allowable Materials | Blackboard Topic Two will include the following: Weekly Readings, Class notes, and Weekly Activities. |
| Required Resources | Web links and example code can be downloaded from the Blackboard portal.  PC with MS Visual Studio, MSOffice. |
| Reasonable Adjustment | In some circumstances, adjustments to assessments may be made for you. If you require support for literacy and numeracy issues; support for hearing, sight or mobility issues; change to assessment times/venues; use of special or adaptive technology; considerations relating to age, gender and cultural beliefs; format of assessment materials; or presence of a scribe you need to inform your lecturer. |
| Assessment Submission | All questions and activities must be attempted. All written answers must be submitted in this assessment document in the appropriate space.  Use of research tools and peers in formulating answers are acceptable – but work submitted must be your own work.  Final portfolio documentation is to be uploaded to the appropriate area in the Blackboard course created for this unit.  If you are marked as NYS (Not Yet Satisfactory) on your first attempt, you will be provided with another opportunity to re-attempt the assessment. |
| Project contents | A project of programming tasks and written questions which should be completed in class and finished in the students’ own time on a weekly basis as per the Delivery and Assessment schedule.   * Question 1 – Service Application Proposal * Question 2 – Interface Design * Question 3 – Class Details * Question 4 – Project Details * Question 5 – Sign Off and Approval * Question 6 – Programming Criteria * Question 7 – Testing Report * Question 8 – Demonstration and Submission |

# Scenario

As the Senior Programmer for CITE Managed Services, your task is to develop a fully functional Drone Service Application for Icarus Pty Ltd who operates a local service and repair company. The application will be used by front desk staff to log drones for service and repair.

## Background Information

The Icarus organisation has a long history for servicing all types of drones and unmanned aircraft, their reputation for quality and discretion has made them the leaders in the service industry. The company offers two categories of service, regular and express. This service is unique because once a service tag is issued the policy is to repair or replace, which ensures all services are completed on time. The staff at Icarus has made the following requests that will need to be included in any solution.

## Application Criteria

The Service Processing Application must use two Queue<T> data structures of a simple class which has the following five attributes: Client Name, Drone Model, Service Problem, Service Cost and Service Tag.

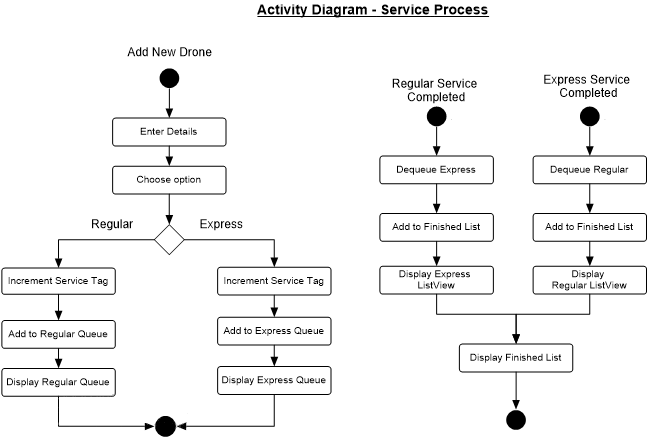
When a client delivers a drone to Icarus for attention the front desk staff will enter the details as required to populate the Drone class. The client will be able to select between a regular or express priority for the service of their drone. The express priority service will incur an additional 15% charge to the service cost. Once the priority has been selected the drone will be added to one of the two queues (regular or express). The client’s drone will be tagged and send to the service department for inspection and repair/service. Once the drone is repaired and returned to the front desk the Icarus staff will remove the details from the queue. This removal process will dequeue the appropriate data structure and add the details to the list of completed work. Finally, the client will be able to pay for the work and collect their drone; the staff on the front desk will then remove the item from the finished work list.

All user interactions must have full error trapping and feedback messaging which is displayed in a status strip at the bottom of the form. The need to use a message box for critical errors or general feedback is not necessary.

You should consult with the CITE representative (Your Lecturer) if you are unsure about any of the problems or questions. Your primary research should focus on the resources on the Blackboard website, additional information can be collected from the Internet, ensure all sources are referenced at the end of your report. You should write your answers in the standard templates provided in this assessment document.

## Application Activity Diagram

The following activity diagram is supplied by the staff at Icarus to highlight the workflow.



## Question 1 Service Application Proposal

Complete the Service Application Proposal template with the project details, then provide a comprehensive list of requirements and features that satisfy the criteria outlined by the staff of Icarus Pty Ltd.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Service Application Proposal | | | | |
| Project Name | | Drone Service Application | | |
| Date | | 12/10/2022 | | |
| Student Name | | Joshua Farrell | | |
| Icarus Staff Requirements | | | | |
| Req. # | Description | | Importance | Notes |
| 1 | A Drone class object is populated with Client Name, Drone Model, Service Problem, Service Cost and Service Tag | | 1 | The attributes must be private with getters and setters. |
| 2 | The Drone object is enqueued into one of two queues, Regular and Express, which is selected by the client | | 3 | Drones within the Express queue incur an additional 15% service cost.  The queues must use a Queue<Drone> type |
| 3 | The Drone object is dequeued from the queue and is populated into a Finished List | | 4 | The Finished List must use a List<Drone> type |
| 4 | The Drone object is removed from the Finished List when the work is paid. | | 5 | The drone is removed by a Mouse\_DoubleClick action |
| 5 | The Drone details must not have any empty attributes | | 2 | The enqueue button method checks for empty attributes first |
| 6 | All user messages will use a statusStrip | | 6 |  |
| 7 |  | |  |  |
| 8 |  | |  |  |
| 9 |  | |  |  |
|  |  | |  |  |
|  | *Add more lines as required* | |  |  |
| User Interaction and Specifications | | | | |
| How will the application behave and what GUI specifications are required. | | | | |
| * The user enters the drone details in the four textboxes | | | | |
| * The service tag is automatically incremented by 10 and can be manually edited by the user using the numeric up/down control. | | | | |
| * The user selects one of two queue options using two radio buttons | | | | |
| * The user clicks the ENQUEUE button and the drone object is added to the correct Queue<Drone> | | | | |
| * Both Queue<Drone> queues are displayed inside the correct ListView (e.g., regular drones are displayed in the regular ListView and vice versa) | | | | |
| * Drones can be removed from both queues using a REMOVE button which will add the drone to the Finished ListBox. | | | | |
| * The user can double click a drone to remove it from the Finished ListBox | | | | |
| *Add more lines as required* | | | | |

## Question 2 Interface Design

Create a detailed GUI Design along with details of all the components using a suitable GUI design software package (design cannot be hand drawn). Ensure your design covers all the features that are outlined by the Icarus Pty Ltd staff. Use the following template to complete this question.

|  |  |  |  |
| --- | --- | --- | --- |
| GUI Design | | | |
| Software Name | Drone Service Application | URL | https://visualstudio.microsoft.com/ |
| Graphical User Interface Layout | | | |
|  | | | |

## Question 3 Class Details

Create a UML class diagram of the Drone data object and then define the data structure for the Queues and List in the target language. Use the following form to complete this question.

|  |
| --- |
| Class Details |
| UML Class Diagram |
|  |
| Data Structure Definitions |
| internal class Drone : IComparable<Drone>  {  private string clientName;  private string droneModel;  private string serviceProblem;  private double serviceCost;  private int serviceTag;    // This class uses separate getter and setter methods which are public  #region Getters and setters  public string GetClientName()  {  return clientName;  }  // Format clientName to Title case (e.g. joshua farrell > Joshua Farrell)  public void SetClientName(string newClientName)  {  TextInfo ti = new CultureInfo("en-AU", false).TextInfo;  clientName = ti.ToTitleCase(newClientName);  }  public string GetDroneModel()  {  return droneModel;  }  public void SetDroneModel(string newDroneModel)  {  droneModel = newDroneModel;  }  public string GetServiceProblem()  {  return serviceProblem;  }  // Format serviceProblem to Sentence case (e.g. propellers are not spinning. > Propellers are not spinning.)  public void SetServiceProblem(string newServiceProblem)  {  string lowerServiceProblem = newServiceProblem.ToLower();  Regex regex = new Regex(@"(^[a-z])|\.\s+(.)", RegexOptions.ExplicitCapture);  serviceProblem = regex.Replace(lowerServiceProblem, s => s.Value.ToUpper());  }  public double GetServiceCost()  {  return Math.Round(serviceCost, 2);  }  public void SetServiceCost(double newServiceCost)  {  serviceCost = newServiceCost;  }  public int GetServiceTag()  {  return serviceTag;  }  public void SetServiceTag(int newServiceTag)  {  serviceTag = newServiceTag;  }  #endregion  // These methods are used to compare between different Drone class objects and to display drone data as a string  #region Utilities  public int CompareTo(Drone d)  {  return droneModel.CompareTo(d.droneModel);  }  public override string ToString()  {  return GetClientName() + "\t\t$ " + GetServiceCost();  }  #endregion  } |

## Question 4 Project Details

Using a CITE approved agile software development methodology, create a project plan. List and describe all the tasks required to complete the development of the Service Application. Create a new GitHub repository and then use the GitHub Project template to answer this question. Complete the following template as evidence of your work.

|  |  |  |  |
| --- | --- | --- | --- |
| Project Details | | | |
| Project Name | Drone Service Application | | |
| Version Number | 0.0.1 | Date | 12/10/2022 |
| Repository Name: | DroneServiceApplication | | |
| URL | https://github.com/joshuafarrell95/DroneServiceApplication | | |
| Initial Project Tasks | Screen Shots | | |
| Repository Details | Local | | |
| Remote | | |

## Question 5 Sign-off and Approval

You will need to arrange for the previous questions to be reviewed by the Lecturer/Assessor for sign off, approval and feedback before you start Question 6.

Your documents will include:

* Question 1 – Service Application Proposal.
* Question 2 – Interface Design.
* Question 3 – Class Details.
* Question 4 – Project Details (screen shots from source control)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Approval (Lecturer/Assessor use only) | | | | |
| Approver Name | Title | Signature | Date | Approved? |
| Stuart | Lecturer |  | 12/10/2022 | Yes |
|  |  |  |  |  |
| Lecturer Feedback | | | | |
|  | | | | |

## Question 6 Programming Criteria

Create and code the Service Application to demonstrate how a collection of information can be stored using a Windows Application. This application will utilise a Queue<T> and a List<T> of a class object**.** Use the hardware and software supplied in the classroom to accomplish the development, debugging and internal documentation of the final Service Application. Use the Version Control System outlined in previous question to manage your code during the development. Your code must adhere to the CITEMS software development standards. (refer http://www.citems.com.au/)

The following programming criteria and features are required, the exact requirements of the Programming Criteria are essential. Any variation from them will need to be corrected in order to achieve a satisfactory performance.

### Addendum

* The input for the Service Tag must be a numeric up/down control. The minimum value is 100, maximum is 900 with increments of 10.
* The input for the Service Priority must be two radio buttons inside a GroupBox. The two values are Regular and Express. This data is not part of the Drone class.
* The input for the Service Problem must be a multi-lined textbox.
* The service items must be displayed in a ListView which displays all the class attributes.
* The finished service items must be displayed in a ListBox which displays the Client Name and Service Cost.

### Programming Criteria

1. Create a separate class file to hold the data items of the Drone. Use separate getter and setter methods, ensure the attributes are private and the accessor methods are public. Add a display method that returns a string for Client Name and Service Cost. Add suitable code to the Client Name and Service Problem accessor methods so the data is formatted as Title case or Sentence case. Save the class as “Drone.cs”.
2. Create a global List<T> of type Drone called “FinishedList”.
3. Create a global Queue<T> of type Drone called “RegularService”.
4. Create a global Queue<T> of type Drone called “ExpressService”.
5. Create a button method called “AddNewItem” that will add a new service item to a Queue<> based on the priority. Use TextBoxes for the Client Name, Drone Model, Service Problem and Service Cost. Use a numeric up/down control for the Service Tag. The new service item will be added to the appropriate Queue based on the Priority radio button.
6. Before a new service item is added to the Express Queue the service cost must be increased by 15%.
7. Create a custom method called “GetServicePriority” which returns the value of the priority radio group. This method must be called inside the “AddNewItem” method before the new service item is added to a queue.
8. Create a custom method that will display all the elements in the RegularService queue. The display must use a List View and with appropriate column headers.
9. Create a custom method that will display all the elements in the ExpressService queue. The display must use a List View and with appropriate column headers.
10. Create a custom keypress method to ensure the Service Cost textbox can only accept a double value with one decimal point.
11. Create a custom method to increment the service tag control, this method must be called inside the “AddNewItem” method before the new service item is added to a queue.
12. Create a mouse click method for the regular service ListView that will display the Client Name and Service Problem in the related textboxes.
13. Create a mouse click method for the express service ListView that will display the Client Name and Service Problem in the related textboxes.
14. Create a button click method that will remove a service item from the regular ListView and dequeue the regular service Queue<T> data structure. The dequeued item must be added to the List<T> and displayed in the ListBox for finished service items.
15. Create a button click method that will remove a service item from the express ListView and dequeue the express service Queue<T> data structure. The dequeued item must be added to the List<T> and displayed in the ListBox for finished service items.
16. Create a double mouse click method that will delete a service item from the finished listbox and remove the same item from the List<T>.
17. Create a custom method that will clear all the textboxes after each service item has been added.
18. All code is required to be adequately commented. Map the programming criteria and features to your code/methods by adding comments above the method signatures. Ensure your code is compliant with the CITEMS coding standards (refer http://www.citems.com.au/).

## Question 7 Testing Report

Once you have completed coding the application, ensure your code is error free and functions correctly. Run the application and check all the UI features work correctly and the various controls are fully functional. You will need to complete the following testing report and provide appropriate evidence that your code functions as expected (references to screen captures). Complete the following Test Report template below.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Testing Report | | | | | | | | | | |
| Project Name | | Drone Service Application | | | | | | | | |
| Methods | | Workflow | | | | | | | | |
| Description | | These tests will test the complete workflow of the Drone Service Application to ensure that the application functions as per client requirements. | | | | | | | | |
| Level of Testing | | Systems testing | | | | | | | | |
| Developer | | Joshua Farrell | | Tester | Joshua Farrell | | Date | | 19/10/2022 | |
| Test Case No | Test Case Name | | Test steps | | | Test Data | | Expected result | | Pass / Fail |
| 1 | Test incomplete drone data | | Add test data, then click on ENQUEUE. Ignore drone model and service cost. | | | ClientName = “joshua farrell”  ServiceProblem= “test” | | Drone is not added to queue, user is warned | | PASS |
| 2 | Test complete drone data and send to Regular Queue | | Add test data as per CSV file in order of service tag, then click on ENQUEUE | | | As per “TestDataReg.csv” | | Drones are added to regular queue | | PASS |
| 3 | Test complete drone data and send to Express Queue | | Add test data as per CSV file in order of service tag, then click on ENQUEUE | | | As per “TestDataExp.csv” | | Drones are added to express queue with 15% added to the serviceCost {115, 172.5, 230} | | PASS |
| 4 | Test select drone data in regular queue and output to text boxes | | Click on test data from “Joe Bloggs” to populate the data. | | | “Joe Bloggs” test data | | Client name and service problem are output to correct textboxes | | PASS |
| 5 | Test select drone data in express queue and output to text boxes | | Click on test data from “Tom Bloggs” to populate the data. | | | “Tom Bloggs” test data | | Client name and service problem are output to correct textboxes | | PASS |
| 6 | Dequeue two drones from regular queue | | Click on REMOVE FROM REGULAR QUEUE | | | N/A | | “Joshua Farrell” and “Joe Bloggs” test data are removed from queue first | | PASS |
| 7 | Dequeue two drones from express queue | | Click on REMOVE FROM EXPRESS QUEUE | | | N/A | | “Jane Citizen” and “Tom Bloggs” test data are removed from queue first | | PASS |
| 8 | Test remove from finished queue | | Double click on “Joshua Farrell” and “Tom Bloggs” to remove from queue. | | | N/A | | “Joe Bloggs” and “Jane Doe” remain in finished queue. “John Citizen” and “Harry Citizen” remain in respective queues | | PASS |
| 9 | Add regular test data with high service tag | | Add line 2 of test data as per CSV file. | | | As per line 2 of “TestDataReg.csv”, except serviceTag = 890 | | Service tag should count to 900. | | PASS |
| 10 | Add express test data with high service tag | | Add line 2 of test data as per CSV file. | | | As per line 2 of “TestDataExp.csv”, except serviceTag = 900 | | Service tag should reset and count to 100. | | PASS |

## Question 8 Demonstration and Submission

Complete the documentation for Questions 6 and 7 then request a suitable time to demonstrate your Service Application for approval. The demonstration will ensure your code is compliant and your documentation conforms to CITE standards. Use the following marking guide to check you have completed all the tasks.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Assessor Marking Guide | | Satisfactory | | Comment |
| **Questions** | | YES NO | |  |
| Q1 | Service Application Proposal is complete. |  |  |  |
| Q2 | Interface Design has software details and layout diagram. |  |  |  |
| Q3 | Class Details has UML class diagram and data structure specification |  |  |  |
| Q4 | Project Details is complete with several screen captures of the VCS. |  |  |  |
| Q5 | Sign-off and Approval has been authorised by the Lecturer. |  |  |  |
| Q6 | Programming Criteria, all the following criteria have been completed  Q6.1 Separate class file |  |  |  |
| Q6.2 List of Class |  |  |  |
| Q6.3 Queue of class (regular service |  |  |  |
| Q6.4 Queue of class (express service) |  |  |  |
| Q6.5 Add new service item |  |  |  |
| Q6.6 Increase express service by 15% |  |  |  |
| Q6.7 Custom method to return radio button priority |  |  |  |
| Q6.8 Custom method to display regular service queue in ListView |  |  |  |
| Q6.9 Custom method to display express service queue in ListView |  |  |  |
| Q6.10 Keypress method for service cost |  |  |  |
| Q6.11 Custom method to increment service tag control |  |  |  |
| Q6.12 Mouse click method to populate textbox from regular service ListView |  |  |  |
| Q6.13 Mouse click method to populate textbox from express service ListView |  |  |  |
| Q6.14 Button method to dequeue regular data structure and add item to list. |  |  |  |
| Q6.15 Button method to dequeue express data structure and add item to list. |  |  |  |
| Q6.16 Double click method to remove item from listbox and list data structure |  |  |  |
| Q6.17 Custom method to clear textboxes |  |  |  |
| Q6.18 Comments mapping criteria to code and all code meets organisational standards |  |  |  |
| Q7 | Testing Report, all UI interactions have been tested. |  |  |  |
| Q8 | Demonstration and Submission |  |  |  |
|  | **Assessment Decision**  Satisfactory  Not Yet Satisfactory | | | |

**Note:** All documentation must use the supplied templates/forms.

**Submit the zipped solution folder with relevant documents to Blackboard**

End of Assessment Task Three