|  |
| --- |
|  |

|  |
| --- |
| **Technical Report** |
| ICTPRG532 – Apply Advanced Object-Oriented Language Skills |
|  |
|  |
|  |
| **Sebastian Vowels** |
| **10/11/2021** |
|  |

Table of Contents

[Task 1 – Email 2](#_Toc87534248)

[Task 2 – Email and Sign-Off Sheet for Presentation of Software Implementation Plan 2](#_Toc87534249)

[Task 3 – Web Application 4](#_Toc87534250)

[Front End (ASP NET Core MVC) 4](#_Toc87534251)

[Task 4 – Web Service 7](#_Toc87534252)

[Back End(ASP.NET Web API) 7](#_Toc87534253)

[Nested Classes in Action 12](#_Toc87534254)

[Task 5 – Drag and Drop and 2-D Graphics 12](#_Toc87534255)

[Sorting Algorithms Comparison 18](#_Toc87534256)

[Searching Algorithms Comparison 19](#_Toc87534257)

[Task 6 21](#_Toc87534258)

[Defect Logs 21](#_Toc87534259)

[Debugging Tools in Action 22](#_Toc87534260)

[Task 7 –Test Data 22](#_Toc87534261)

[Task 8 –Test Report 22](#_Toc87534262)

[Test Summary 22](#_Toc87534263)

[Task 10 - Documentation 23](#_Toc87534264)

[Documentation Maintenance 23](#_Toc87534265)

[Task 11 – Sign-off Sheet 24](#_Toc87534266)

# Task 1 – Email

Graphical user interface, text, application, email

Description automatically generated

# Task 2 – Email and Sign-Off Sheet for Presentation of Software Implementation Plan

Graphical user interface, text, application, email

Description automatically generated

|  |  |  |  |
| --- | --- | --- | --- |
| **Project Name:** | Fantastic Feedback – Web Application and Service | | |
| **Project Manager:** | Shaun O’Sullivan | | |
| **Start Date:** | 1/09/2021 | | |
| **Completion Date:** | 12/11/2021 | | |
| **Project Deliverables:** | | | |
| Deployed Web Application | | | |
| Deployed Web Service | | | |
| Documentation | | | |
| Network Surveyor Java Application | | | |
|  | | | |
|  | | | |
|  | | | |
|  | | | |
|  | | | |
|  | | | |
|  | | | |
|  | | | |
| ***I acknowledge that I have submitted all the stated deliverables as per the Project Requirements and your instructions.*** | | | |
| Sebastian Vowels | |  | 10/11/2021 |
| **Programmer’s Name** | | **Signature** | **Date** |
|  | |  |  |
| **Project Manager** | | **Signature** | **Date** |

# Task 3 – Web Application

## Front End (ASP NET Core MVC)

Graphical user interface, text

Description automatically generated

Figure - GitHub Desktop Application Showing Use Of Git In Project

A picture containing text, electronics, computer, screenshot

Description automatically generated

Figure - FFFrontEnd APIRequest Showing XML comment blocks and use of naming conventions

Chart, line chart

Description automatically generated

Figure - Firefox Showing Rendered Web Page

A picture containing text, monitor, indoor, black

Description automatically generated

Figure - FFFrontEnd Startup, appsettings, APIRequest Showing the process of subscribing to the back-end

A screenshot of a computer

Description automatically generated with medium confidence

Figure - FFEFrontEnd Using NuGet Package Manager to Obtain Third-Party Library for the Solution

Graphical user interface, text

Description automatically generated

Figure - FFFrontEnd SurveyController and APIRequest Showing Code to Obtain Data From Web Service

# Task 4 – Web Service

## Back End(ASP.NET Web API)

A screenshot of a computer

Description automatically generated with medium confidence

Figure - API-FantasticFeedback SurveyController Showing the GetAll method

A screenshot of a computer

Description automatically generated with medium confidence

Figure - API-FantasticFeedback SurveyController Showing the GetSurveyDetails method

A screenshot of a computer

Description automatically generated with medium confidence

Figure - API-FantasticFeedback SurveyController Showing the GetSingleSurvey method

A screenshot of a computer

Description automatically generated with medium confidence

Figure - API-FantasticFeedback SurveyController Showing the PostSingleSurvey Method

A picture containing text, monitor, computer, electronics

Description automatically generated

Figure - API-FantasticFeedback SurveyController Showing the DeleteSingleSurvey Method

A computer screen capture

Description automatically generated with medium confidence

Figure - API-FantasticFeedback SurveyController Showing the UndeleteSingleSurvey Method

A picture containing text, monitor, electronics, screenshot

Description automatically generated

Figure - API-FantasticFeedback SurveyController Showing the PutSingleSurvey Method

A screenshot of a computer

Description automatically generated with medium confidence

Figure - API-FantasticFeedback FFAPIContext Class Showing the Model

A computer screen capture

Description automatically generated with medium confidence

Figure - API-FantasticFeedback appsettings Showing the Database Connection Strings

## Nested Classes in Action

A screenshot of a computer

Description automatically generated with medium confidence

Figure - Apache NetworkSurveyor BinaryTree class showing Node class

# Task 5 – Drag and Drop and 2-D Graphics

Graphical user interface, application

Description automatically generated

Figure - Firefox Showing Drag and Drop Feature in Web Application

A computer screen capture

Description automatically generated with medium confidence

Figure - FFFrontEnd Create Survey View Showing Dropzone Code

A screenshot of a computer

Description automatically generated

Figure - Firefox Showing 2-D Graphics On FrontPage of Web Application

A picture containing text, monitor, screenshot, computer

Description automatically generated

Figure - FFFrontEnd Showing Index View and HomeController Code that Displays 2-D Graphics

Graphical user interface, text, application

Description automatically generated

Figure - Firefox Showing Help Page and Link Visible In Nav Bar on All Pages

A computer screen capture

Description automatically generated with medium confidence

Figure - Apache NetBeans Showing Code That Interacts with LinkedList, BinaryTree, and Hashing Classes Within Application

A screenshot of a computer

Description automatically generated with medium confidence

Figure - Apache NetBeans Showing Class Responsible for Creating and Maintaining Doubly Linked List

A computer screen capture

Description automatically generated with medium confidence

Figure - Apache NetBeans Showing Class Responsible for Creating and Maintaining Binary Tree

A screenshot of a computer

Description automatically generated

Figure - Apache NetBeans Showing Method In Application That Interacts With HashMap

## Sorting Algorithms Comparison

There are many sorting algorithms available to utilise, the following are examples of just three:

* Bubble Sort
  + A bubble sort is a relatively simple sorting algorithm. A dataset is iterated through comparing the current item with the next item. If the current item is larger than the next item, the two swap positions in the dataset, otherwise the next item becomes the current item and the process repeats. This iteration is done as many times as the length of the dataset. This way it ensures that every relevant comparison is conducted.
* Selection Sort
  + A selection sort is another relatively simple sorting algorithm. Within a dataset, the smallest (or largest, depending on the manner in which the dataset is to be sorted) is found and placed at the start of the dataset. This process then repeats for the second smallest, third, etc. Until the entire dataset is organised. The selection sort is less demanding of the computer as less comparisons are made and as a result can be faster than a bubble sort.
* Insertion Sort
  + An insertion sort operates in a similar manner to the selection sort, however it differs in that it chunks the dataset. Instead of following the same principle of a selection sort for the entire dataset, it breaks it up into smaller pieces and iterates through them. Once this is completed, your dataset is not sorted, but you have individual chunks that have been pre-sorted for you. The same selection sort principal is repeated but is more effective as the data is somewhat organised already.

Within the application all three were implemented into the program, an example of the Bubble Sort implementation is shown below.

A screenshot of a computer

Description automatically generated

Figure - Apache NetBeans Showing Bubble Sort Method

## Searching Algorithms Comparison

There are many search algorithms available to utilise, the following are examples of just three:

* LinkedList Find()
  + The find method implements a search algorithm by starting at the head of the linked list and then proceeding through the list in order. Each node provides the details of the next node in the list. Upon arriving at the next node in the list, the algorithm examines if the current nodes data matches the desired data. If it does, the algorithm returns the nodes details, otherwise it continues through the list.
* BinaryTree inOrderTraversal()
  + The inOrderTraversal() method implements a search algorithm by utilising recursion within the code. Within the method, there are three steps: navigate to left node, return node value, navigate to right node. Starting at the root node of the tree and then navigates to the left node of the current one and calls the same method for the new node. The new method has the same instructions and as such performs the same operations. Eventually there wont be a new node to traverse too, in which case the left most node will be recorded. After this the right node will be traversed too. What is most beneficial about this approach is that as the methods complete (all three instructions completed), the tree begins to get traversed upwards.
* Linq .where()
  + Linq operations are used for searching and sorting data. The .where() method pulls data out of a collection where the data matches the specified input. For example, Surveys.where(c => c.SurveyID == 2) will return the Survey from the Surveys table that has an id of 2. This algorithm is built into the .NET framework and is widespread in its use.

All of the three search algorithms were utilised in the application, an example of the Linq.where() algorithm is shown below:

Text

Description automatically generated

Figure - API-FantasticFeedback Showing use of Linq quieries, including .Where(), inside SurveyController

# Task 6

Graphical user interface, text

Description automatically generated

Figure - API-FantasticFeedback Showing Breakpoints and Watches Being Implemented

## Defect Logs

|  |  |
| --- | --- |
| **Syntax / Logic Errors** | **Solutions Applied** |
| Dropzone styling in create survey view not as intended | Inspected view through browser tools to determine what styling was being applied to the div. Determined that incorrect css file was used and called in wrong order. |
| FrontEnd calls to API were being rejected based on 404 errors | Stepped through API and found that no endpoints were being activated. Discovered that variables were being passed through but no endpoints accepted the required variable. Implemented variable in API endpoints. |
| Was unable to obtain string from session data through known methods | Read documentation and found a method that is known to work with session data, implemented it and working as intended. |
| Implementation of Chart.Js wasn’t working as intended. | Sought help from colleagues who identified missing code from view. Implemented ideal code, and feature working as intended. |
| Context not returning correct value from database. | Wrong table being accessed, Survey should’ve been Surveys. Feature working as intended. |

## Debugging Tools in Action

Breakpoints can be set by clicking in the left-hand column of the line you wish to set it. A red dot will appear indicated that the breakpoint has been set. Watches can be created during debugging by right clicking on the desired variable within the code and clicking “add watch”.

Running the application in debug mode will allow breakpoints to be hit if the code reaches the specified line. Watches are only relevant when they are currently being utilised by memory.

# Task 7 –Test Data

Please refer to the adjacent document “*3\_Test Report Template Functionality.xlsx*” and refer to sheet “*TEST CASES List*”. This document detailed the testing performed on this application.

# Task 8 –Test Report

Please refer to the adjacent document “*3\_Test Report Template Functionality.xlsx*” and refer to sheet “*Test Report*”. This document details the results of implementing the aforementioned tests.

## Test Summary

High Priority:

* Test: 7, Section: 3
  + The Question GET endpoint for the API returned all questions within the DB, not just those attached to the users account. Could be related to previous test result (9,1). No Authorisation built into endpoint.
* Test: 13, Section: 3
  + The Option GET endpoint for the API returned HTTP status code 405. This indicates that the specified endpoint does not exist. Specified endpoint was: "localhost:xxxxx/api/Option/{Username}".
* Test: 14, Section: 3
  + The Option GET {id} returned question that was requested. However as a result of test 7,3 one tester decided to use an option that was not attached to the authenticated account. They should not have been able to retrieve this option from the API.

Moderate Priority:

* Test: 9, Section: 1
  + The tester was able to navigate to a FE webpage that returned all questions with the database. Potentially linked to findings from test 7,3.
* Test: 10, Section: 1
  + The tester was able to navigate to a FE webpage that returned all options within the database. Potentially linked to findings from test 13,3.
* Test: 2, Section: 5
  + Throughout the testing any errors returned to the user through the FE interface were evaluated for the quality of information provided to the user and if logging was accurately put in place for later investigation. One error from the FE was encountered. The information returned to the user was not acceptable and no logger was found for this error. See Test Results for further details.

Overall the functionality of the project is at an acceptable standard. The key issues presented relate to Authorisation features lacking from the API as well as error handling and logging lacking. Please review the Test Results document for a more in-depth overview of the findings.

# Task 10 - Documentation

### Documentation Maintenance

The application will be documented after production and will continue to be updated throughout any future developments.

The application version will be numbered with minor updates/bug updates to cause an increment of 0.1 and major updates/new feature inclusions will cause the increment to increase by 1.

The project documentation and application will be packaged in a zip folder and uploaded to the connect system to be stored securely there. A backup will be stored within my cloud based storage services. The naming conventions will be as outlined below.

Within this organisation the naming standards are as follows:

[my surname]\_[my student number]\_[unit number]\_[project name]. This is the standard that is expected within the organisation and as such it is the one I will utilise.

Throughout the development of the application GitHub, and Git, will be used to maintain version control. The Git version control system allows the developer to create commits to the relevant development branch. This allows them to use multiple devices to continue working on code as well as ensuring that the code will be up to date with the latest build each time work continued.

# Task 11 – Sign-off Sheet

|  |  |  |  |
| --- | --- | --- | --- |
| **Project Name:** | Fantastic Feedback | | |
| **Project Manager:** | Shaun O’Sullivan | | |
| **Start Date:** | 1/09/2021 | | |
| **Completion Date:** | 12/11/2021 | | |
| **Project Deliverables:** | | | |
| Deployed Web Application | | | |
| Deployed Web Service | | | |
| Documentation | | | |
| Network Surveyor Java Application | | | |
|  | | | |
|  | | | |
|  | | | |
|  | | | |
|  | | | |
|  | | | |
|  | | | |
|  | | | |
| ***I acknowledge that I have submitted all the stated deliverables as per the Project Requirements and your instructions.*** | | | |
| Sebastian Vowels | |  | 10/11/2021 |
| **Programmer’s Name** | | **Signature** | **Date** |
| Shaun O’Sullivan | |  |  |
| **Project Manager** | | **Signature** | **Date** |