T-LEVELS | Institute for Apprenticeships & Technical Education

Lead Examiner Report

Summer 2023

T Level Technical Qualification in Digital Production, Design and Development- Employer Set Project

Pearson

Introduction

The tasks in the Employer Set Project (ESP) follow a standardised format, and the expectations in terms of individual task outcomes continues to be consistent with all previous assessed materials. These previous materials include both the sample assessment material and the additional sample assessment materials, as well as the previous live series assessments all of which are published on the Pearson website.

This series saw a slight drop in the mean mark achieved by students in comparison to the two most recent series. However, there was also a significant increase in the number of students entered, and the number of providers who had entered students for the first time. Analysis of data shows that, broadly speaking students from returning providers, i.e. providers that have entered other cohorts of students previously, tended to perform better on the assessment. Typically these students were more aware of the required processes and tended to produce better organised and more effective outcomes. All providers, both new and returning, are encouraged to use previous assessment materials to help prepare students for the assessment.

The ESP assessment is made up of five tasks (Task 1, 2, 3, 4a and 4b), requiring students to demonstrate knowledge of a range of specification topics, demonstrate practical skills and apply understood skills to a given scenario.

Performance in individual tasks is covered in detail in the next section of this report, but broadly speaking while more consistent outcomes have been seen in some areas (such as the Gantt chart) there are still areas where significant improvement is required. In particular, the rationale in Task 1 and Tasks 4a and 4b.

Individual questions

The following section considers each question on the paper, providing examples of popular student responses and a brief commentary of why the responses gained the marks they did. This section should be considered with the live external assessment and corresponding mark scheme.

Task 1

This task requires students to interpret a substantial brief for an overarching project and develop a resource management and financial plan that can meet the requirements of the project. The task requires students to interpret a range of data, including revenue and costs, and to forecast the project's development as well as ongoing benefits of the project.

Students are provided with a range of tasks that are required to be completed during the project, as well as profiles for members of staff that can work on the project.

Students are then required to provide a rationale that should explain and justify the decisions they made.

Gantt-

While the mean mark for this task did not vary significantly it was observed by examiners that overall students' outcomes in this task were much more consistent. Students typically were able to allocate suitable workers to task and the order of tasks was usually logical. Weaker students do tend to still arrange these in the order listed in the task document with minimal consideration of how these tasks may be better organised to mimic a real-life project. For example, the back-end database would need to be completed before other units that rely on data such as the data analysis module and the stock control module.

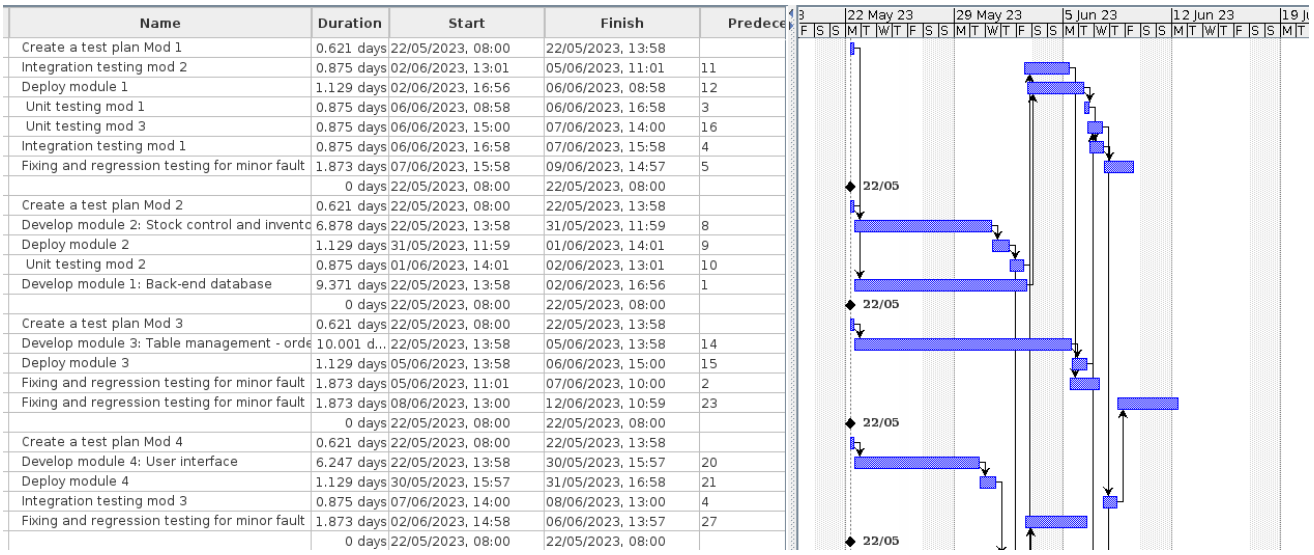
Stronger examples of Gantt charts showed a clear understanding of the relationship between tasks, as well as how concurrent running of tasks, clear milestones and suitable allocation of staff demonstrates a better understanding of the project development process.

An improvement was seen in the clarity of the assignment of the resources to the individual tasks. In many cases this may be down to the increased use of project planning software (e.g. Projectible, MS Project) while the use of project planning software is not required, and some very good Gantt charts that used Excel were seen, the tools and features provided by this type of software can aid students by speeding up the process allowing them more time on the other 2 parts of this assessment task. The benefits of the Project software also include automatic cost

calculations and highlighting where they may have been mistakenly assigned the same resource twice, which can be of great benefit to the students during the pressure of a timed examination. These tools also make it easier for students to move tasks and resources around when considering different ways to ensure the project is planned effectively.

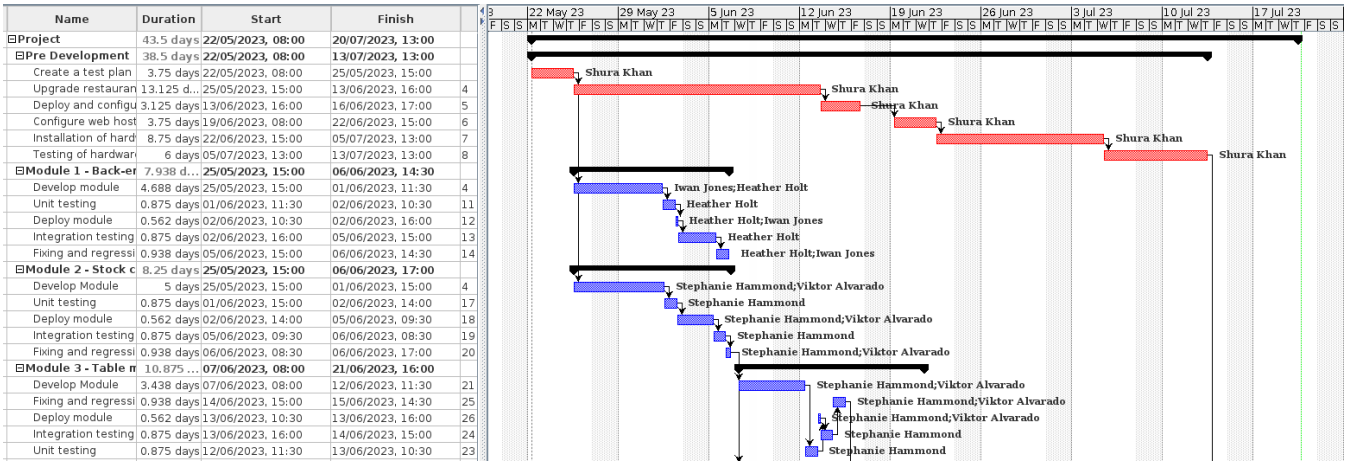
One key area for development in this part of the task is the application of efficient SDLC models. More effective students were able to break the given tasks down into smaller 'sprints', which would also be closely linked to when testing is scheduled. Less effective students tended to place testing of all modules at the very end of the project which does not reflect modern computing projects.

In this example we can see an extract from a mark band 1 response. The student has attempted to consider some appropriate timings and has considered the need for concurrent tasks. However, in the full plan all modules were developed simultaneously, which isn't sensible, as some modules would be dependent on others. Also they have not allocated any staff to the tasks, so it is not possible to see if the tasks are assigned appropriate resources:



Name	Duration	Start	Finish	Predece
Create a test plan Mod 1	0.621 days	22/05/2023, 08:00	22/05/2023, 13:58	
Integration testing mod 2	0.875 days	02/06/2023, 13:01	05/06/2023, 11:01	11
Deploy module 1	1.129 days	02/06/2023, 16:56	06/06/2023, 08:58	12
Unit testing mod 1	0.875 days	06/06/2023, 08:58	06/06/2023, 16:58	3
Unit testing mod 3	0.875 days	06/06/2023, 15:00	07/06/2023, 14:00	16
Integration testing mod 1	0.875 days	06/06/2023, 16:58	07/06/2023, 15:58	4
Fixing and regression testing for minor fault	1.873 days	07/06/2023, 15:58	09/06/2023, 14:57	5
	0 days	22/05/2023, 08:00	22/05/2023, 08:00	
Create a test plan Mod 2	0.621 days	22/05/2023, 08:00	22/05/2023, 13:58	
Develop module 2: Stock control and inventc	6.878 days	22/05/2023, 13:58	31/05/2023, 11:59	8
Deploy module 2	1.129 days	31/05/2023, 11:59	01/06/2023, 14:01	9
Unit testing mod 2	0.875 days	01/06/2023, 14:01	02/06/2023, 13:01	10
Develop module 1: Back-end database	9.371 days	22/05/2023, 13:58	02/06/2023, 16:56	1
	0 days	22/05/2023, 08:00	22/05/2023, 08:00	
Create a test plan Mod 3	0.621 days	22/05/2023, 08:00	22/05/2023, 13:58	
Develop module 3: Table management - orde	10.001 d...	22/05/2023, 13:58	05/06/2023, 13:58	14
Deploy module 3	1.129 days	05/06/2023, 13:58	06/06/2023, 15:00	15
Fixing and regression testing for minor fault	1.873 days	05/06/2023, 11:01	07/06/2023, 10:00	2
Fixing and regression testing for minor fault	1.873 days	08/06/2023, 13:00	12/06/2023, 10:59	23
	0 days	22/05/2023, 08:00	22/05/2023, 08:00	
Create a test plan Mod 4	0.621 days	22/05/2023, 08:00	22/05/2023, 13:58	
Develop module 4: User interface	6.247 days	22/05/2023, 13:58	30/05/2023, 15:57	20
Deploy module 4	1.129 days	30/05/2023, 15:57	31/05/2023, 16:58	21
Integration testing mod 3	0.875 days	07/06/2023, 14:00	08/06/2023, 13:00	4
Fixing and regression testing for minor fault	1.873 days	02/06/2023, 14:58	06/06/2023, 13:57	27
	0 days	22/05/2023, 08:00	22/05/2023, 08:00	

In this example the student was place in mark band 3:



Name	Duration	Start	Finish	
Project	43.5 days	**22/05/2023, 08:00**	**20/07/2023, 13:00**	
Pre Development	38.5 days	**22/05/2023, 08:00**	**13/07/2023, 13:00**	
Create a test plan	3.75 days	22/05/2023, 08:00	25/05/2023, 15:00	
Upgrade restauran	13.125 d...	25/05/2023, 15:00	13/06/2023, 16:00	4
Deploy and configu	3.125 days	13/06/2023, 16:00	16/06/2023, 17:00	5
Configure web host	3.75 days	19/06/2023, 08:00	22/06/2023, 15:00	6
Installation of hard	8.75 days	22/06/2023, 15:00	05/07/2023, 13:00	7
Testing of hardwar	6 days	05/07/2023, 13:00	13/07/2023, 13:00	8
Module 1 - Back-er	7.938 d...	**25/05/2023, 15:00**	**06/06/2023, 14:30**	
Develop module	4.688 days	25/05/2023, 15:00	01/06/2023, 11:30	4
Unit testing	0.875 days	01/06/2023, 11:30	02/06/2023, 10:30	11
Deploy module	0.562 days	02/06/2023, 10:30	02/06/2023, 16:00	12
Integration testing	0.875 days	02/06/2023, 16:00	05/06/2023, 15:00	13
Fixing and regressi	0.938 days	05/06/2023, 15:00	06/06/2023, 14:30	14
Module 2 - Stock c	8.25 days	**25/05/2023, 15:00**	**06/06/2023, 17:00**	
Develop Module	5 days	25/05/2023, 15:00	01/06/2023, 15:00	4
Unit testing	0.875 days	01/06/2023, 15:00	02/06/2023, 14:00	17
Deploy module	0.562 days	02/06/2023, 14:00	05/06/2023, 09:30	18
Integration testing	0.875 days	05/06/2023, 09:30	06/06/2023, 08:30	19
Fixing and regressi	0.938 days	06/06/2023, 08:30	06/06/2023, 17:00	20
Module 3 - Table m	10.875 ...	**07/06/2023, 08:00**	**21/06/2023, 16:00**	
Develop Module	3.438 days	07/06/2023, 08:00	12/06/2023, 11:30	21
Fixing and regressi	0.938 days	14/06/2023, 15:00	15/06/2023, 14:30	25
Deploy module	0.562 days	13/06/2023, 10:30	13/06/2023, 16:00	26
Integration testing	0.875 days	13/06/2023, 16:00	14/06/2023, 15:00	24
Unit testing	0.875 days	12/06/2023, 11:30	13/06/2023, 10:30	23

While this example could still be improved by breaking the development down into smaller chunks, and threading testing throughout the development of each module, they showed a good understanding of the need to test each module at the time of development and the importance of where possible scheduling concurrent tasks.

Plan for the resources selected and associated costs-

This task requires students to take into consideration a number of factors including:

- The set costs for the project such as company base fee, hardware for the project and infrastructure upgrades.
- Varying and ongoing project costs, for example in this series students had to choose the type of cloud server package that they would include and the cost of web hosting and advertising.
- Costs based on planning decisions made- for example the final total wage cost would vary depending on which member of staff is assigned to which task and for how long.
- How the cost of the project will impact on the company over time. This should include consideration of the company's current annual revenue, their current annual outgoings and the impact on future finances based on the projected change in revenue.

In general, students did not perform particularly well on this task. While most were able to access some marks, typically students only produced responses that accessed mark band 1 or the bottom of mark band 2 (1-2 marks).

Typically, the financial planning provided by students did not consider planning decisions made. For example, wage totals were often derived through simple calculations of *total project hours * each individual's pay rate* rather than considering the number of hours a particular staff member worked on the project. This is again where the use of project planning software would be of great benefit to students, as students would be able to add the members of staff and their hourly rate to a resource list, and the software will automatically calculate the totals based on the number of hours each member of staff is assigned. This information can then be easily transferred to their financial planning spreadsheet.

Another kay area for development in this task is the consideration of the impact of the predicted revenue increase. Students often did not integrate the current outgoings into the forecast of future profit/loss or did not consider the impact that making a low in Year 1 would have when carried over into year 2.

In this example we can see an example of a response that was placed at the top of mark band 2 (3 marks). The response is mostly correct. Most calculations are correct and there is consideration of impact over 3 years, however the Project Lead's pay did not match the plan they had produced and as such the final total calculation was out by quite a significant value.

Company Financial Data		
Cost	Per unit	Total Amount (£)
Forecast for development		
Purchasing of hardware (per restaurant)	2150	10750
Training for restaurant staff		2300
Software development company base fee		43000
Option 1 Server (per month)	235	2820
Upgrades to company infrastructure (per restaurant)	4725	23625
Project lead's pay (per day)	175	5250
Sr. Software Engineer's pay (per hour)	75	11325
Jr. Software Engineer's pay (per hour)	45	6570
Web Development Engineer's pay (per hour)	57	7011
Hardware and Networking Technician's pay (per hour)	62	15376
Database Engineer's pay (per hour)	70	22610
Additional annual costs		
Web hosting and online advertising (per month)	825	9900
IT Support Technician		19500
Total		180037

	Annual revenue	Annual cost	Additional costs	Net income
Year 0 (Development)	1560000	1357200	180037	22763
Year 1 (+15%)	1794000	1357200	32220	404580
Year 2 (+17.5%)	1833000	1357200	36300	439500
Year 3 (+12%)	1747200	1357200	36300	353700

Rationale-

This part was the poorest performing part of Task 1. Typically, students were not able to move beyond mark band 1 (1-3 marks out of 9). Responses mostly provided a narrative of what they had done or provided only very superficial reasoning as to planning decisions made.

Responses often just described who had been assigned to what task and what order the tasks had been placed in. this information is already provided within the Gantt chart and planning document, so a narrative does not provide any additional information. Similarly, where justification was provided by students it often did not move beyond stating that jobs had been allocated based on the staff member's job title.

The purpose of the rationale is for students to demonstrate a deeper understanding of the planning process and explain their reasoning for any decisions they made. Some areas that could be considered, although they are not limited to these, may include:

- Explaining why they have chosen to break a task down in to smaller parts, why one task has been scheduled before another, or why they have scheduled different forms of testing at different stages in the project.
- Providing justification as to why some tasks may have more than one member of staff allocated, e.g. placing a more senior member of staff to oversee a junior/less experience member of staff particularly at the start of the project when they are unproven.
- Potential risks that have been identified for example in this project the cost of the project combined current annual costs means that very

little may be made in profit, and any unforeseen costs may cause them to lose profit. In terms of staffing Viktor, the junior software engineer, only had 12 months of experience so may need additional support or extra time to complete tasks. Iwan the web development engineer was new to the company and has only previously work on small scale projects.

Task 2

Task 2 requires students to demonstrate their knowledge of testing and debugging code. To do this, the students are given a piece of non-functioning code, a test plan template, and the requirements of the program.

Students are required to debug and test the code, to implement a functioning solution and effectively document the testing/debugging process.

While there are still some important areas for improvement in this task, it should be said that there were some very positive outcomes seen in this task during this series.

Typically, students tended to perform better at the actual process of fixing the code, compared to the documentation of the process.

Many students were able to successfully fix most defects within the provided cost, and there was a clear improvement on the number of students that were able to fix issues beyond those that would be identified by the IDE. However, there were pockets of students that although they identified an error flagged by the IDE and attempted to fix the identified error, it was clear that they did not further test the correction that they had implemented. A common issue was how students corrected the error with the dictionary.

Here we can see an example where the student has made changes so that it doesn't flag an error in the IDE when it is first run. But the change is not functional and causes the program to crash later.

```
6   menu = (
7       ("Nachos" == "5.50"), ("Soup" == "4.95"),
8       ("Burger" == "10.50"), ("Brisket" == "12.50"), ("Ribs" == "15.00"),
9       ("Corn" == "2.50"), ("Fries" == "3.00"), ("Salad" == "3.25")
10  )
```

The correct fix should have been:

```
4   menu = {
5       "Nachos": 5.50 , "Soup":4.95,
6       "Burger":10.50, "Brisket": 12.50, "Ribs":15.00,
7       "Corn":2.50, "Fries":3.00, "Salad":3.25
8   }
```

While most students were able to demonstrate how to test if the code was functional and shower some understanding of testing for valid and invalid inputs, test data was often quite limited. Student testing often showed a lack of understanding between of the differences between invalid and erroneous. As a

result, appropriate data was often not selected to support the stated test.

Testing documentation also failed to consider extreme data (that is data that falls at the extremes of a boundary) and so often logic was not adequately tested, and errors persisted.

For this specific scenario suitable boundary testing would have included:

- Ensuring the correct table number was entered and used by:
  - Using the values 0 and 1 to test the lower boundary.
  - Using values 9, 10 and 11 to test the upper boundary.

Many students also failed to adequately test the calculations in the program. When documenting testing, students are encouraged to provide tests that would highlight potential issues in calculations.

Here for example, we can see an extract from a students work that provides a range of test data that has been selected to test the bill output, the student has clearly documented the error that has been found, how they will fix it and the results of the retest.

	1, Steve, 10, Burger, 1, Ribs, 1, Fries, 2, X, 2, 10, 1	**The program outputs a bill**	**The program outputs a bill, but gives an incorrect subtotal** Subtotal before discount = £37.5 Final total after discount = £-525.0	**The final_total calculation on line 190 multiplies the subtotal by the discount and then subtracts that from the subtotal. So in this case it is '37.5 – (37.5*15)', which doesn't calculate the percentage correctly. To fix this, I changed the discount numbers from 15 & 25 to 0.15 and 0.25 on lines 126 and 128**
	1, Steve, 10, Burger, 1, Ribs, 1, Fries, 2, X, 2, 10, 1	**The program outputs a bill**	**Outputs the bill correctly and the order processing system is printed, but the decimal places are not in line with the 2 decimal places that currency normally has** TABLE 10 - Server name: Steve Summary of bill before discounts: ['Burger', 1, 10.5, 'Ribs', 1, 15.0, 'Fries', 2, 6.0] Discount percentage applied to this bill = 0.15 Subtotal before discount = £37.5 Final total after discount = £31.875	**I would need to round up the float to two decimal places for both the subtotal and the final_total, but I don't remember how to do that. I also added some spacing around the bill to make it more apparent where it is within the program.**

This could be further improved by including a manual calculation in the 'expected outcome' column which can be compared against the actual result produced by the code.

Task 3

This task requires students to consider a given system requirement, and related data to produce a set of design documents that could be used to produce an appropriate solution.

Many students did quite well on this task. Often students showed a good understanding of the program requirements and were able to provide a design for a solution either as Pseudocode or as a flow chart.

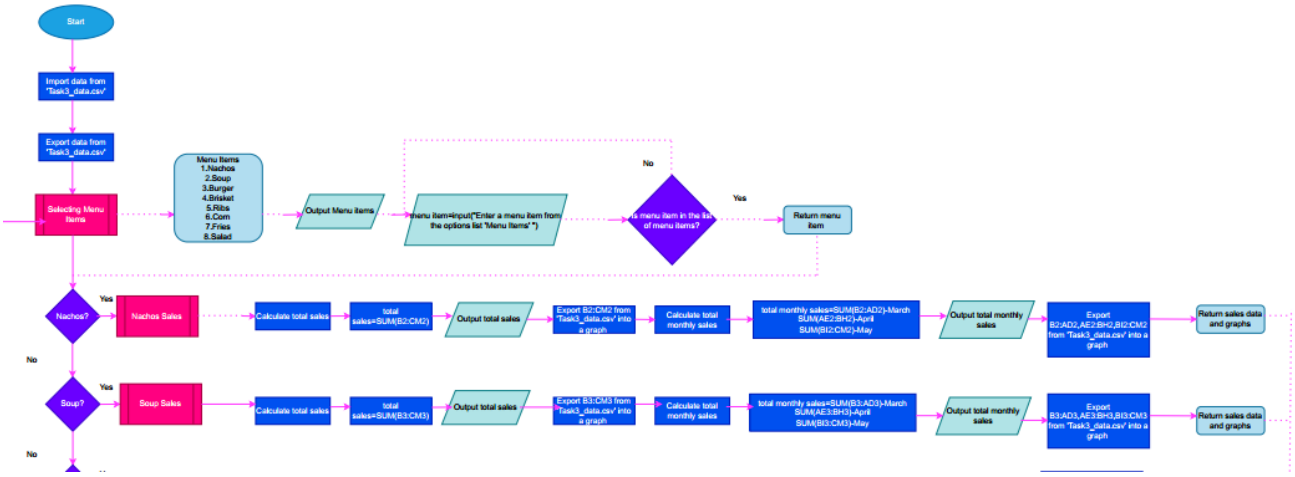
Where students did less well this was often due to limited understanding of the given requirements. The example data for this task is provided as a .csv file which students are encouraged to explore in detail, alongside the given requirements before starting their design.

Many students provided a decomposition diagram, although this is not a specific requirement of the task, students were given credit as appropriate. However, due to the time demands of the assessment they are encouraged to demonstrate decomposition through their chosen algorithms rather than through a decomposition diagram so that they can spend more time producing the requested outputs. This can be achieved by breaking their proposed solution down in to relevant sections, for example in their Pseudocode they should demonstrate how the solution would be broken down into appropriate functions.

In this task specifically decomposition for key process as callable functions could include:

- Getting and validating user input for choice of menu item.
- Importing the data set and storing it in a suitable runtime data structure.
- Getting and validating user input for start and end date for data range.
- Generating data subsets/additional data frames based on user inputs (e.g. menu item and start date etc.).
- Generating and displaying a graph of selected data.
- Calculating totals or averages over a given time period to discern ‘popularity.’

Here we seen an example extract from a student's work:



Things the student has done well:

- Shown some decomposition of the problem into smaller processes (importing data, getting, and returning menu options).
- Covers some key inputs, processes, and outputs (e.g. user selections, calculating and outputting averages and totals).
- Mostly appropriate flowchart conventions applied.

Things that could be improved:

- Clarity of communication- presenting the solution as a large single flowchart is not full appropriate.
- Make use of more programmatic conventions. For example, the flowchart uses spreadsheet style formula which is not fully appropriate.
- Repetition- solution is quite repetitive; it would be better to show a single reusable process as would be appropriate for a coded solution.

Task 4a

This task was quite poorly answered by many students. Often the code provided was not functional and contained significant issues in relation to meeting the given requirements. Many leaners seemed to be unfamiliar with the areas of coding that are required to complete this task which resulted in many students gaining very few marks in this task.

To improve in this area providers are encouraged to work with students in using the libraries Pandas and MatplotLib so that they are able to create code that can:

- Import data from a provided .csv file and hold it in a data frame for further processing.
- Create and manipulate data frames which utilise all or part of the imported data (as required to meet given requirements).
- Perform mathematical operations and statistically analysis on the contents of a data frame (or associated variable) such as:
    - Identifying trends and patterns over time
    - Calculating a total or average from a range of data
    - Counting the number of occurrences of a specific item of data
- Output data and results in a range of formats (e.g. text, tables and graphs)

One area that has improved slightly in this series was in the clarification of a key misconception. Previously many students produced a 'log in' function which they added to the given code, in an attempt to meet the 'security' assessment focus requirements. Whilst the number of students doing this has reduced, it is clear that some providers are still teaching this. It must be stressed that this is NOT what is expected and often this provided code provided little in the way of appropriate functionality for the user. Focusing on this detracts students from spending time on creating code that would meet the given requirements.

To address the requirements of the 'security' assessment focus students should demonstrate secure coding and data practices. Securing considerations in relation to secure coding can be demonstrated through:

- Avoiding global variables and passing appropriately between functions.
- Avoiding the use of a single generate data frame to ensure security of the data instead it is good practice to generate a new data frame within a function.
- Error handling: If the system crashes, is any data returned in the error message?

Task 4b

Student performance in this task continues to be very poor. Most students were not able to provide appropriate reflective evaluation of the outcomes from Task 4a.

Responses were often descriptive in nature and provided a narrative of what had been created with little consideration as to how these outcomes met the given program and user requirements.

Providers are encouraged to work with students to develop their evaluative skills. Students should be aware that an evaluation required them to make a value judgement in relation to criteria (i.e. the given requirements). While it is good practice for students to include examples from their work, they should not just describe the code, but should instead provide a detailed exploration of how the code that they have produced meets those requirements.

Students should then draw upon their evaluation, and their understanding to the brief, to make suggestions on how the code could be improved. It should be stressed that while some improvements might e for things that they feel they did not successfully complete, they should be considering things that they have completed successfully but could be refined. For example, they may explore how they interpreted the requirement of 'trends and patterns over time for lunch and dinner service'. They may have successfully included a facility to allow the user to input a start and end date but may consider how it may improve user experience to providing a clearer comparison of the two services using graphs, or even calculations to summarise differences in totals or averages.

Any suggestions for improvements or additional functionality should be meaningful and have a direct relevance to the given scenario.

Summary

Based on the performance on this paper students should focus on:

- Effective application of project management models by breaking project tasks down into shorter sprints.
- Providing more effective rationales for planning decisions made by demonstrating a greater awareness of the scenario and applying this to explore a range of positive and negative implications.
- More effective use of test data, including use of boundary testing to check logic and checking accuracy of calculations.
- During testing in Task 2 ensuring any changes made to the code are fully tested and this process is documented in detail.
- Ensuring greater facility with Pandas and MatplotLib to be able to develop code to meet the given requirements.
- Developing higher quality evaluations (Task 4b). responses should focus on the quality of what was produced and not just provide a narrative of the process.