Cloud Computing (COMM034)

Coursework description, 2022-23

Contents

[Nature of coursework – it is vital you have read and understood this section 3](#_Toc125548234)

[Aim 4](#_Toc125548235)

[Submission 4](#_Toc125548236)

[Deadline 4](#_Toc125548237)

[Relationship to learning outcomes 4](#_Toc125548238)

[A note on linearity of description 4](#_Toc125548239)

[Approach 5](#_Toc125548240)

[Which data will your system analyse? 7](#_Toc125548241)

[VM setup and code for the approach 8](#_Toc125548242)

[Requirements – see the user scenario, below, as will support explanation of these 10](#_Toc125548243)

[Brief example user scenario 11](#_Toc125548244)

[Submissions 12](#_Toc125548245)

[Weighting and Composition 13](#_Toc125548246)

[Marking criteria 14](#_Toc125548247)

**THE POSTING OF THIS DOCUMENT ON ANY** WEBSITE WITHOUT EXPLICIT PERMISSION HAVING FIRST BEEN OBTAINED FROM THE MODULE LEADER **IS AN INFRINGEMENT OF COPYRIGHT AND COULD RESULT IN LEGAL ACTION**.

THIS DOCUMENT MUST NOT BE SHARED WITH **ANYBODY** OUTSIDE THE UNIVERSITY OF SURREY.

# Nature of coursework – it is vital you have read and understood this section

This is an **individual** coursework. It is **not** a group coursework. It is your own individual efforts that will gain marks for you. Efforts of others on your coursework may become a problem for both you and them, and it is best to avoid this.

You may *discuss* what you are doing with others on the module. BUT do not *share* report content or code with each other. Also, do not copy+paste material from elsewhere even if you are then changing some of the words of what you have pasted. To use what others have written directly, quote their text properly using “”, attribute it to them, and then discuss or interpret what is important about it for your purpose. Similarly, the source of images that you have not created but are using must be stated. Note that any direct inclusion of work by others implies that marks for what is included should be given *to them* –only your efforts gain marks for you. Similar applies to code also.

**As students from the recent past will be able to testify, unhappily, inappropriate copying can become problematic for progression through the programme.**

Note that software tools are in use in respect to detection:

1. **Turnitin** will detect copying from elsewhere *as well as* from each other – for the latter, this information only becomes available after all submissions have been received.
   1. Note that there is no ‘magic number below which everything is acceptable’ in Turnitin. The nature of what is being identified is important.
   2. Assessors may also recognise similar patterns between submissions where Turnitin does not, or know source materials that Turnitin appears not to. Turnitin is not exhaustive, and you should avoid underestimating the assessors!
2. **Code similarity software** is also used to detect copying between your code submission and that of others, with a focus on code added to that already provided in the module.

You should be aware, also, that the module leader has several research publications in plagiarism detection and is one of the Department’s Academic Integrity Officers.

For more information see, for example: <https://exams.surrey.ac.uk/academic-integrity-and-misconduct/plagiarism> : “Failure to understand what constitutes plagiarism, pressure of time, or conflicting deadlines for assessed work are not acceptable as sufficient explanations for the submission of plagiarised material.”

# Aim

To demonstrate your understanding of how to critically explain, and construct, a Cloud application using multiple services across Cloud providers, involving user-specifiable scaling.

You will explain, implement, evaluate, and demonstrate,

**an application that supports determining the risks of using certain trading signals for a trading strategy** using a so-called **Monte Carlo** method.

*Such an application might support, for example, building a trading strategy that uses* trailing stops *– though you are not expected to do this. Background reading may, however, help:* [*https://www.investopedia.com/articles/trading/08/trailing-stop-loss.asp*](https://www.investopedia.com/articles/trading/08/trailing-stop-loss.asp)

Your application will need to adopt the **Approach** within the set of provided **Requirements**.

# Submission

Submissions will be made via SurreyLearn and comprise two components:

1. A **PDF** document of a **maximum of 4 pages** that conforms to the template provided
2. A **Zip** file for code for your system - **excluding** libraries that will merely bloat the zip file.

Reminder: this is an **individual** coursework and must not be worked on in pairs or groups.

*Submissions* ***will*** *be evaluated using tools including Turnitin.*

# Deadline

Submissions are to be made by the deadlines given for the module’s Assignment folders on SurreyLearn. *Standard University penalties apply to late submissions.*

# Relationship to learning outcomes

|  |  |
| --- | --- |
| **LO** | **Description** |
| LO1: | Demonstrated with regard to the selected Cloud services and the software implementation, and related questions of cost and performance. |
| LO2: | Demonstrated with regard to use of Google App Engine and Lambda, as well as justification of the second scalable service, within an industrial/academic problem context. |
| LO3: | Demonstrated with regard to evaluation of alternative and/or additional services and appropriateness of elaboration of the system overall. |
| LO4: | Demonstrated with regard to defining the system in the context of Cloud. |
| LO5: | Demonstrated through the specification, design, implementation and critical evaluation of the software implementation. |

# A note on linearity of description

Some aspects are described across passages and sections – e.g. Audit. Prior to posing questions, check that you have seen/searched all such mentions within the document first.

# Approach

*There are no marks available for re-explaining this approach in your submission.*

**Note that a core of Python code is provided for this approach, within this document, and this Python code must be used for the application created.**

1. The approach involves identifying trading signals in financial time series and capturing the risk associated to these. Such an assessment might support a subsequent evaluation of a trading strategy.
   1. Financial time series here comprise daily data – specifically, a summary of the trading day comprising the Open/High/Low/Close values for each trading day (OHLC values can readily be produced for other time intervals – e.g. every 15 minutes).
   2. OHLC data can be visualised using a “Japanese candlestick”, and certain resulting shapes interpreted from these to indicate something about the data that may ‘signal’ making a trade (buy or sell). The figure below is an example, using real data, of such candlesticks where:
      1. Open and Close values provide the top and bottom of the ‘box’ on each candlestick – if Close is higher than Open, price movement was upwards overall and the body is **green** (a price rise from the start of the day to the end); Open higher than Close and the body is **red** (a price fall from the start of the day to the end); other charts and some pattern naming might use **white** for upward and **black** for downward or other colour/shading schemes.
      2. A line projecting from the top of the body indicates that the High was above the respective Open/Close; a line projecting from the bottom of the body indicates that the Low was below the respective Open/Close. Such lines are referred to as the wick or shadow.

Chart, box and whisker chart

Description automatically generated

* + 1. Resulting shapes can have various names such as a Green/Red Marabozu (Japanese for dominance) or Spinning Top, and can involve more than one candlestick – for example Harami (Japanese for pregnant) or Three Black Crows. Note how the latter name implies a specific colour/shade scheme.

Here, we’re only going to look at 2 patterns that each involve 3 candlesticks: (i) Three White Soldiers; (ii) Three Black Crows. For the above figure, the combination of 6th , 7th and 8th candlesticks from the left should fit our expectations for Three Black Crows, and **example code is provided that will act as a detector for such shapes in data being obtained directly from Yahoo Finance** (the pattern identification code could be more sophisticated, for example it doesn’t check if the 2nd and 3rd have Open value between Open and Close – the real body – of the candle before, but we’re using simpler principles here).

|  |  |
| --- | --- |
| **Three White Soldiers** – rising, close values above previous close values and each close above the open | Three white soldiers |
| **Three Black Crows** – falling, close values below previous close values and each close below the open | Three black crows |

Table 1: Images from <https://www.ig.com/uk/trading-strategies/16-candlestick-patterns-every-trader-should-know-180615>

* 1. What we want to know, before we might conduct any other analysis, is how much **risk** would be associated to each potential signal and whether each might generally be **profitable** some days after the signal. For risk, a Monte Carlo analysis offers one option: we use characteristics of the recent price history to simulate a substantially longer price series, then determine the amount that might be lost and confidence involved – for example, this could be expressed between people as: “there is a 95% confidence that no more than 5% of the amount traded would be lost, and a 99% confidence that no more than 7.5% of the amount traded would be lost”:
     1. If we have a minimum price history requirement of 101 days, including the signal, we first calculate the daily returns – the % change in value compared to the day before, i.e. (price-previous)/previous – which would offer 100 such values.
     2. The returns series will be characterised by its *mean* and *standard deviation*, and we use a random number generator (Normal/Gaussian distribution) to simulate (generate) a series containing lots of such values that could closely fit to these parameters.
     3. By sorting the resulting series, of potential gains and losses, and picking off values at 95% and 99%, we know theoretical % changes of interest with respect to what could be expressed between people. We could, by extending from this point, use these values to see if there are ways to optimize the trading strategy – for example, by trading in high, or low, risk situations.
     4. **Example code is provided** that offers an example of capturing such values. This analysis needs to use high numbers of ‘shots’, but this takes time - and *each user is impatient*. It is quite possible to undertake such analysis using parallel resources: each resource generates a new series and provides its values; these values are then averaged, appropriately, in order to generate the resulting two values needed. .

To determine if it’s profitable, we simply choose how many days later we want to look at price difference.

**For this coursework, we want a system that will reduce the overall wait time for results for a larger value at risk analysis (more shots) and enable the user to identify which signals relate more, or less, risk and whether profitable.**

## Which data will your system analyse?

**Your** system will use one of the “Other symbols” identified in the code comment depending on the 2nd character of your Surrey username (i.e. a username such as q**q**0134):

1. If the second character is from ‘**a**’ to ‘**f**’ inclusive: **ZM**
2. If the second character is from ‘**g**’ to ‘**l**’ inclusive: **AMZN**
3. If the second character is from ‘**m**’ to ‘**r**’ inclusive: **BP.L**
4. If the second character is anything else: **NFLX**

# VM setup and code for the approach

First, locate and start the COMM034 VM (check that the network connection has not become lost!)

In the VM, create a requirements.txt file listing the following 3 libraries (known to work at the versions listed – at least yfinance version 0.2.14 seems to believe symbols don’t exist!) and install using pip3, per Lab 1:

|  |
| --- |
| pandas==1.4.1  yfinance==0.1.70  pandas\_datareader==0.10.0 |

*Warnings such as “Can't uninstall 'pytz'. No files were found to uninstall.” can be safely ignored.*

Note that if you wanted to add them to a GAE project you would extend the requirements.txt for that – i.e. ensuring that Flask, gunicorn, boto3 are already included.

The code provided on the next pages should, for initial testing, be put into a single file. This provides for the core of the code that would be expected to be seen. However, it will be expected that you will ‘take this apart’ to use in your system.

|  |
| --- |
| #!/usr/bin/env python3  import math  import random  import yfinance as yf  import pandas as pd  from datetime import date, timedelta  from pandas\_datareader import data as pdr  # override yfinance with pandas – seems to be a common step  yf.pdr\_override()  # Get stock data from Yahoo Finance – here, asking for about 3 years  today = date.today()  decadeAgo = today - timedelta(days=1095)  # Get stock data from Yahoo Finance – here, Gamestop which had an interesting  #time in 2021: https://en.wikipedia.org/wiki/GameStop\_short\_squeeze  data = pdr.get\_data\_yahoo('GME', start=decadeAgo, end=today)  # Other symbols: TSLA – Tesla, AMZN – Amazon, ZM – Zoom, ETH-USD – Ethereum-Dollar etc.  # Add two columns to this to allow for Buy and Sell signals  # fill with zero  data['Buy']=0  data['Sell']=0  # Find the signals – uncomment print statements if you want to  # look at the data these pick out in some another way  # e.g. check that the date given is the end of the pattern claimed  for i in range(2, len(data)):  body = 0.01  **# Three Soldiers**  if (data.Close[i] - data.Open[i]) >= body \  and data.Close[i] > data.Close[i-1] \  and (data.Close[i-1] - data.Open[i-1]) >= body \  and data.Close[i-1] > data.Close[i-2] \  and (data.Close[i-2] - data.Open[i-2]) >= body:  data.at[data.index[i], 'Buy'] = 1  #print("Buy at ", data.index[i])  **# Three Crows**  if (data.Open[i] - data.Close[i]) >= body \  and data.Close[i] < data.Close[i-1] \  and (data.Open[i-1] - data.Close[i-1]) >= body \  and data.Close[i-1] < data.Close[i-2] \  and (data.Open[i-2] - data.Close[i-2]) >= body:  data.at[data.index[i], 'Sell'] = 1  #print("Sell at ", data.index[i])  # Data now contains signals, so we can pick signals with a minimum amount  # of historic data, and use shots for the amount of simulated values  # to be generated based on the mean and standard deviation of the recent history  minhistory = 101  shots = 10000  for i in range(minhistory, len(data)):  if data.Buy[i]==1: # if we’re interested in Buy signals  mean=data.Close[i-minhistory:i].pct\_change(1).mean()  std=data.Close[i-minhistory:i].pct\_change(1).std()  # generate much larger random number series with same broad characteristics  simulated = [random.gauss(mean,std) for x in range(shots)]  # sort and pick 95% and 99% - not distinguishing long/short risks here  simulated.sort(reverse=True)  var95 = simulated[int(len(simulated)\*0.95)]  var99 = simulated[int(len(simulated)\*0.99)]  print(var95, var99) # so you can see what is being produced |

For your system, you will need to take the code above **and build from it** appropriately along with what you have seen in labs to meet the set of Requirements provided.

Note, in particular, that little of the code above needs to run in parallel, and consider where code that needs only to be run once per session should be run, and when, within the system.

**Additional advice (a small hint):** it is readily possible to avoid using libraries (e.g. Pandas) that will otherwise take more effort to use in the scalable services (Lambda, EC2 etc) – and it is advisable to avoid needing those as it can take more effort to get them to work there (esp. in Lambda)!

If you wanted to avoid having to support use of DataFrame in Lambda, look at what, for example,

[entry[3] for entry in data.values.tolist()] might offer with respect to the above – and that it is possible to go further still.

# Requirements – see the user scenario, below, as will support explanation of these

1. You must use: (i) Google App Engine, (ii) AWS Lambda, **and** (iii) one of the other *scalable services in AWS*: Elastic Compute Cloud (EC2), Elastic MapReduce (EMR) or – should you wish to explore – EC2 Container Service (ECS).

Subsequent mentions of *scalable* *services* in this document mean **Lambda plus your choice** from EC2 or EMR or ECS.

1. Your system must offer a persistent front-end through which the user will be able to *initialise* (create or ‘warm up’, as necessary) and *terminate* (as necessary to remove any possible continuous cost) scalable services separately from using them to run the analysis.
2. The *scalable* *services,* and not Google App Engine, must calculate risk values – Google App Engine can be used to collect and average risk values.
3. The system must provide for the following with respect to how to initialise and analyse:
   1. Initialisation:
      1. A way for the user to specify which of *your* two scalable services, as **S**, to use for estimating – i.e. if you have chosen EC2, the selection here will be *between* Lambda and EC2. Note that, for this example, only Lambda or EC2 would then be being used for analysis unless the user changes to the other scalable service;
      2. A way for the user to specify a value of **R**, as the number of resources (in the *scalable* *services*) to be used in parallel for calculating risk;
      3. Using **S** and **R**, a way to provision (create or ‘warm up’, as necessary) the appropriate number of resources in the scalable services (note that ‘warm up’ is needed for all of the scalable services). This is likely to include readying any other data or service connections needed in advance of any analysis;
      4. Capture of the time required for creation or ‘warm up’ such that it could be reported to the user and is available for analysis with respect to overall system running costs.
   2. For the risk analysis –the system must provide the following:
      1. A way to specify the value of **H** as the length of price history from which to generate the mean and standard deviation;
      2. A way to specify the value of **D** as the number of data points (**shots**) to use by each **R** for calculating risk;
      3. A way to specify the value of **T** as Buy or Sell to allow for separate analysis of each type of signal;
      4. A way to specify the value of **P** as the number of days after which to check profit (or loss);
      5. Using **H**, **D**, **T**, and **P**, a way to run the calculations across the **R** resources where each resource returns its own risk values for averaging, and information is captured about the runtimes involved, such that the work done can be reported to the user and be stored for analysis – see ‘Audit’ page, below.
   3. For output – the system must provide the following:
      1. A result page with (a) a chart, using either Image Charts or the [old] Google Chart service, showing a line each for the 95% and 99% risk values for each signal and two lines relating the averages over each such that higher and lower risk signals can be seen readily, and (b) a table where each row shows signal date, associated risk values, and profit/loss values.
      2. An ‘Audit’ page, showing information about each selection of **S, R, H, D, T, P,** the value of total profit (or loss), the two averaged risk values, and the compute runtime/cost for all analysis undertaken to date - such that you could use this information to estimate costs for much higher numbers of data points (**D**).
   4. Reset – the system must provide a way to ‘zero’ the analysis without needing to warm up new resources.
   5. Switch off – the system must provide a way to ‘terminate’ EC2/EMR/ECS resources so that no further costs would be incurred.

Your system *may* incorporate additional Cloud components, for example for storage for the ‘Audit’. However, the mantra of Keep It Stupid-Simple should be followed and additional components should not be added unnecessarily.

# Brief example user scenario

The user asks the system to ‘warm up’ 4 resources (**R**) of the type selected (**S**). Resources, whether Lambda, EC2, EMR, or any other, are brought to a point where they are ready for running analysis.

The user specifies 80,000 shots (**D**) per **R**, with a history of 200 days per signal (**H**) and for Buy signals (**T**) and a Profitability time horizon **P** of 10 days.

In doing so, the user expects that, **for each signal**, 320,000 shots are being produced in total, and there will be the averaging of 4 values for each of 95% and 99% - i.e. two values per R – that results in one value per signal for each of 95% and 99% - and only these **two values result from analysis per signal**. In addition, the value of profit (or loss) for each signal is calculated using the difference between the price at the signal and, when available, the price P days forward of the signal – for a Buy signal, there is profit if the price has moved higher but loss if lower; for Sell, there is profit if the price has moved lower but loss if it has moved higher.

Following this analysis, the user will be presented with a chart showing all risk values – the two risk values for each signal, and two lines of averages over these – one over the 95% signal values and one over the 99% signal values. The total value of profit/loss, and the table, will also be presented to the user.

The user could run further analysis, but for this scenario has done enough so wants (non-Lambda) resources to be terminated. This does not, however, delete the Audit, which needs to be stored across uses/sessions (NB: variables within Python code do not allow for this over extended time periods, and nor do other temporary storage mechanisms that depend on continued running of supporting components).

# Submissions

The document **must** use the coursework template. Template settings for margins, font sizes, font and paragraph spacing, and so on **must not be altered**. Doing so will impact on structure/quality, and quality forms part of the overall assessment; your submission may also be pasted into the original template and checked for length (see above).

Numbers of pages stated relates an absolute maximum – for the avoidance of doubt, anything appearing on pages 5 and beyond will not be marked.

**Two** columns must be used throughout with only large figures and tables allowed to span two columns.

Nothing stated in the coursework template is to be taken to contradict this coursework description in any way. Furthermore, only the latest available coursework description must be used, along with any communication posted on SurreyLearn about corrections or alterations.

**Submit a written paper, of a maximum of 4 pages, using the template provided, that presents the system. As well as providing a Title, your name, URN, email address and a link to your front-end, this needs to:**

1. Provide an abstract and introduction which comprehensively and very clearly relates your system to NIST SP 800-145 with respect to i) what the developer uses/experiences and ii) what the user uses/experiences. Assume that you would write this as though using Real AWS, even though you are using AWS Educate, and so there would be no restrictions on what would be accessible.

*Note that simply stating or attempting to rewrite NIST SP-800-145 is not sufficient for this.*

1. Present (i) a high-level view of the roles/responsibilities of the major components of your system and (ii) discuss what data/information needs to be communicated between these major components in order for the system to work and meet the requirements. A figure can be included to support this – only one such figure should be included – provided that it uses the AWS architecture icons with suitable additions. Explain, also, (a) your choice for the second *scalable service* over the other possibilities, and (b) briefly, why you chose any other services to add and what alternatives you considered but dismissed and the reasons used.

*Note that this means explaining what would be expected to be done, and where, within your system – so this does* ***not*** *require: (i) explaining any of the cloud services or how they might be generally used or useful; (ii) explaining how to use any of the cloud services; (iii) explaining code or technical mechanisms as relates to implementation; (iv) describing what the user does. Explanations that are not required will occupy space in the submission without attracting marks.*

1. Clarify the extent to which requirements were MET (M), PARTIALLY MET (P), or NOT MET (N), briefly justifying your selection of those MET or PARTIALLY MET and explain what you could have done in order to meet those PARTIALLY MET and those NOT MET. Satisfaction of requirements is to be cross-referenced to a description of code used from elsewhere, and what you have added to this, such that it is possible to see how the submitted code is related.
2. Present and discuss a set of results that the system produces – show, and clearly explain, what happens to risk values and the time taken for analysis when the numbers of shots/resources used is increased systematically. This section can include, at most, one screenshot from the system showing one set of results (since the URL of your system is to be provided). *Such a screenshot must be clearly legible at 100% zoom.*
3. Explain all of the costs (i.e. over all services) that would be involved with running the system for a large amount of such analysis - these will be ‘real world’ costs, i.e. assuming all free tier usage is exhausted. Costs should relate only to the region used for each service in use.
4. Demonstrate quality in presentation through submission structure, including the abstract and introduction, and quality of writing.

# Weighting and Composition

Coursework represents **100%** of the assessment for this module, and so should be considered as requiring **significant effort** in order to complete.

Weighting for each of the above components of the submission is shown in the Marking Criteria table in the next section.

# Marking criteria

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Mark** | **Intro** | **Architecture** | **Satisfaction of Requirements** | **Results** | **Costs** | **Writing** |
| *Marking Weight* | 20% | 15% | 35% | 10% | 10% | 10% |
| **80-100** | Excellent demonstration of relating to NIST 800-145, demonstrating mastery of applicability. | Exemplary discussion of how the system is composed and operates. Easy to conceive of the implementation from this point. | Complete system, exquisitely critiqued, and/or with significant discussion of possible improvements and complete clarity over code use and additions. | Excellent discussion of results from the system, demonstrating substantial depth of exploration. | Comprehensive treatment of costs. | Highest standard of submission with logical progression and clear discussion. A well-written and highly readable piece with negligible spelling or grammatical errors that provides an effective overview. |
| **70-79** | High standard of demonstration of relating to NIST 800-145, demonstrating a good extent of mastery in respect to relevant components. | High standard of presentation of how the system is composed and operates. Easy to conceive of the implementation from this point, although specific aspects will require clarification. | Mostly complete system, well-critiqued with significant discussion of possible improvements.  Effective clarification of code. Detail used in an effective manner to ensure illustration of salient features. | Effective discussion of results from the system demonstrating significant depth of exploration. | High standard of treatment of costs. | Clear and logical progression and clear discussion. An engaging, well-written and highly readable submission with negligible spelling or grammatical errors that provides an effective summary. |
| **60-69** | Demonstration of clear associations to NIST 800-145, showing a reasonable extent of mastery in respect to relevant components. | Relatively comprehensive standard of presentation of how the system is composed and operates. Specific aspects will require additional clarification. | Robust efforts made towards completing a system; critique and discussion of possible improvements exists but could be more considered. Detail used in varying amounts to illustrate the salient features. | Evidence of analysis of data by the system and how to obtain with some exploration. | Consideration of costs has specific gaps or limitations. | Clear progression and good standard of discussion. Structure has room for improvement and spelling and grammatical errors *may* distract the reader. |
| **50-59** | Showing some clarity with respect to NIST 800-145, but not necessarily relating. The extent of such coverage also requires some improvement to ensure that correctness or coverage convinces others. | Good demonstration of how the system is composed and operates, but vital aspects of the system are less clearly outlined. | Moderate efforts made towards completing a system; some moderate critique and discussion of possible improvements exists. Detail demonstrates how the system should work, but is unconvincing. | Evidence of the generation of results yet discussion, requires improvement. | Consideration of costs requires improvement in investigation. | Clarity of progression and standard of discussion has good room for improvement. Spelling and grammatical errors may distract the reader. |
| **40-49** | An unclear demonstration of how NIST 800-145 relates to the system, either due to limited discussion or limited understanding. | Certain key aspects of the composition of the system are omitted, which suggests difficulty in the implementation. These should have been addressed as a priority. | Parts of the system may exist; some critique and discussion of possible improvements exists, but in a limited form.  Clear that rather more implementation effort was needed. | Little evidence that the approach to analysis was sufficiently understood such that the system could be used to produce useful results, and discussion is limited. | Limited consideration of costs. | Difficulties in clarity. Standard of discussion has significant room for improvement. Spelling and grammatical errors may significantly distract the reader. |
| **30-39** | Limited or inadequate knowledge/association presented. | Inherently limited system description. | Limited evidence towards satisfying requirements and/or any related critique and discussion. Difficult to discern what may be being claimed. | Barely a hint of evidence of analysis and implementation leading to possible results. | Very limited consideration of costs. | Poor standard of presentation. More attention needs to be paid to producing work of a suitable quality in future. |
| **< 30** | No demonstration that the student has gained useful knowledge of NIST 800-145. | Inadequate or no demonstration that the student has considered how to construct such a system. | Inadequate or no evidence that the student knows what to do to satisfy requirements. | Inadequate or no evidence that the student knows what to do. | Inadequate or no treatment of costs. | Very poor standard of presentation. |