Digital Solutions Summative external assessment (EA): Examination (25%)

|  |
| --- |
| Unit 4: Digital impacts   * Topic 1: Digital methods for exchanging data * Topic 2: Complex digital data exchange problems and solution requirements * Topic 3: Prototype digital data exchanges |
| Cognitions assessed (5 out of 8):   1. recognise and describe programming elements, components of exchange systems, privacy principles and data exchange processes 2. symbolise and explain programming ideas, data specifications, data exchange processes, and data flow within and between systems 3. analyse problems and information related to a digital problem 4. ~~determine solution requirements and criteria~~ 5. synthesise information and ideas to determine possible low-fidelity components of secure data exchange solutions 6. ~~generate components of the digital solution~~ 7. evaluate impacts, components and solutions against criteria to make refinements and justified recommendations 8. ~~make decisions about and use mode-appropriate features, language and conventions for particular purposes and contexts~~ |
| **Time**: 2 hours plus perusal (15 minutes)  **Length**: 800–1000 words in total, including   * 50–250 words for short-response answers * 400 words or more for the extended response |
| * Section 1 multiple choice [*10 marks*] * Section 2 short response [6*0 marks*] * Section 3 extended response [3*0 marks*]   Total 100 marks |

# Section 1

## Instructions

* Answer all questions in the question and response book.
* This book will not be marked.
* This section has ten questions and is worth 10 marks.

**Question 1**

Identify a one-way method of encrypting plain text to a fixed length digest so that it can’t be (or is extremely difficult to be) decoded:

1. public / private key encryption
2. asymmetric encryption
3. hashing
4. XML

**Question 2**

A web developer tests their new e-commerce “shopping cart” web application using the following tests:

|  |  |
| --- | --- |
| Test # | Test Description |
| 1 | Check if the webpage works with an assistive text-to-speech reader for users with visual difficulties |
| 2 | Check that the shopping cart icon hyperlink can be found in the top-left corner of the page by a first-time user |
| 3 | Check that an item can be removed from a shopping cart |
| 4 | Check that a ‘timeout’ or failed transaction does not result in the customer being charged for the purchase, nor does it mark a transaction as being complete |
| 5 | Check with both customer and client that the completed application provides the ability for their business to retail their products online |

In order of tests 1-5, the primary *useability principle* being tested by each test is:

1. learnability, safety, utility, effectiveness and accessibility
2. accessibility, utility, effectiveness, safety and learnability
3. safety, learnability, accessibility, learnability and utility
4. accessibility, learnability, utility, safety and effectiveness

The algorithm below is used for both Question 3 and Question 4.

**BEGIN**

x = 1

y = 2

z = y + x

**WHILE** (y <= z)

**IF** (x > y) **THEN**

**PRINT**("A")

**ELSE**

**IF** y == z **THEN**

**PRINT**("B")

**ELSE**

**PRINT**("C")

**ENDIF**

**ENDIF**

y = y + 1

**ENDWHILE**

**END**

**Question 3**

Identify the correct output of this algorithm.

1. ABC
2. BC
3. CB
4. CBA

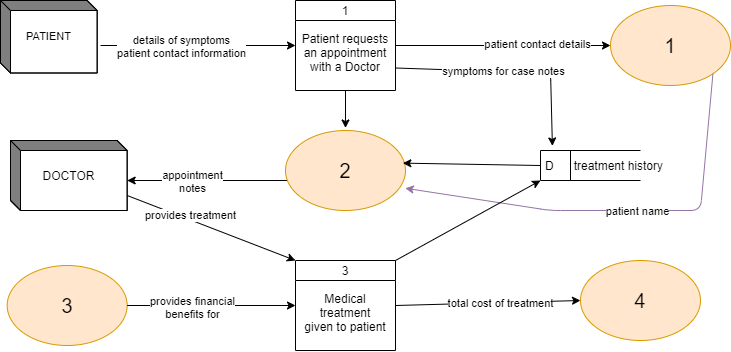
**Question 4**

Which statement best describes the execution pattern of this algorithm?

1. The algorithm does not use any iteration constructs
2. The variable y is used as a counter in the loop
3. The variable x sets the upper limit of the loop
4. The algorithm does not use any selection or branching constructs

**Question 5**

Consider the following data flow diagram.



Select the description that best fits the missing four key elements in the data flow diagram:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **1** | **2** | **3** | **4** |
|  | External entity | Data Store | Process | Data store |
|  | Medical benefits fund | Patient data store | Assign appointment to patient | Billing data store |
|  | Data store | Process | External Entity | Data store |
|  | Billing data store | Medical benefits fund | Patient data store | Assign appointment to patient |

**Question 6**

Consider the sample tables below.

Result table from query

|  |  |  |
| --- | --- | --- |
| title | category | price |
| Sprite | Drinks | 2.5 |
| Hot Dog | Hot Food | 4.5 |

Extract of the food table

|  |  |  |
| --- | --- | --- |
| title | category | price |
| Nachos | Hot Food | 6 |
| Pie | Hot Food | 4 |
| Chips | Hot Food | 3 |
| Hot Dog | Hot Food | 4.5 |
| Sprite | Drinks | 2.5 |
| Apple | Fruit | 1 |
| Fanta | Drinks | 2 |
| Kombucha | Drinks | 6 |

Analyse the food table above to determine the SQL statement that produced the result table shown.

|  |  |
| --- | --- |
| (A) | **SELECT** title, category, MAX(price) AS price  **FROM** food  **WHERE** price < 5  **GROUP BY** category  **HAVING** MAX(price) > 2 |
| (B) | **SELECT** title, category, MAX(price) AS price  **FROM** food  **WHERE** price < 5 OR price > 2  **GROUP BY** category |
| (C) | **SELECT** title, category, MAX(price) AS price  **FROM** food  **GROUP BY** category  **HAVING** MAX(price) < 5 |
| (D) | **SELECT** title, category, MAX(price) AS price  **FROM** food  **WHERE** price > 2  **GROUP BY** category  **HAVING** MAX(price) < 5 |

**Question 7**

Consider the algorithm below.

**BEGIN**

**SUB1**(i)

**BEGIN**

t = 2

c = 1

y = 0

**WHILE** i <= t

y = SUB2(c)

**PRINT**(y)

i = i + 1

**ENDWHILE**

**END**

**SUB2**(i)

**BEGIN**

i = i \* 2

**RETURN** i

**END**

x = -1

SUB1(x)

**END**

Evaluate the algorithm by desk checking and determine the correct output.

1. 0,1,2,3
2. 2,2,2,2
3. 2,4,8,16
4. -2,4,-8,16

**Question 8**

Consider the following fragment of pseudocode.

**BEGIN** main()

**BEGIN** sort(items)

**SET** ordered = FALSE

**WHILE** NOT ordered

**SET** ordered = TRUE

**SET** counter = 0

**WHILE** counter < *length*(items)

**IF** items[counter] > items[counter+1] **THEN**

**SET** ordered = FALSE

**END IF**

**SET** counter = counter + 1

**ENDWHILE**

**ENDWHILE**

**RETURN** items

**END**

**SET** list = [ 5,4,1,8,7 ]

**PRINT**( sort(list) )

**END**

**SET** temp = items[counter]

**SET** items[counter] = items[counter+1]

**SET** items[counter+1] = temp

Which of the following statements is most correct?

1. The pseudocode contains assignments, selections and iterations, but no modularisation.
2. The three lines of pseudocode outlined in the entire rectangle above that swaps a pair of elements in a list could have all been replaced with these two lines of pseudocode:

**SET** items[counter] = items[counter+1]

**SET** items[counter+1] = items[counter]

1. This sort function would not work for a list that has two of the same element, for example: list = [ 1,1,2,2,1 ]
2. The sort function could be used to produce this output: [ 'a','b','c','g','i' ]

**Question 9**

A data dictionary for a table named ‘Printers’ is shown below.

|  |  |  |
| --- | --- | --- |
| Field Name | Data Type | Purpose |
| IP | varchar | IP address that uniquely identifies a printer connected to the network |
| Cartridge\_replaced | datetime | Timestamp that indicates when last cartridge change took place |
| Sheets\_printed | integer | Number of A4 sheets that have been printed. This is reset to 0 when a printer cartridge is replaced |
| Toner\_consumed | decimal (real) | Percentage (given as a decimal, e.g. 0.5 = 50%) of toner consumed. This is reset to 0.0 when a printer cartridge is replaced |

The following query is designed to return data from the Printers table.

**SELECT** IP, Sheets\_printed \* Toner\_consumed **AS** Ratio

**FROM** Printers

**WHERE** Sheets\_printed > (**SELECT** AVG(Sheets\_printed) **FROM** Printers)

Which option best identifies the function of the query?

1. Calculate the ratio of sheets printed to toner used for all printers.
2. Calculate the ratio of sheets printed to toner used for the printer that has printed the most sheets.
3. Calculate the ratio of sheets printed to toner used for the printer that has printed an average number of sheets.
4. Calculate the ratio of sheets printed to toner used for the printer that has printed more than the average number of sheets printed.

**Question 10**

The two algorithms below are designed to ask the user to enter a plain text word, apply a Caesar shift to the word, and print out the resulting cipher text. For example, inputting the plain text “ABC” should produce the cipher text “DEF”. The Caesar shift should use a base of 26, so that ciphered characters ‘wrap’ to fit within the range of alphabet characters available. For example, inputting the plain text “XYZ” should produce the cipher text “ABC”.

The two algorithms were evaluated against the criteria of accuracy and safety.

|  |  |
| --- | --- |
| Term | Explanation |
| Accuracy | the condition or quality of being true, correct, exact, precise or correct |
| Safety | ability for users to make errors and recover from the mistake |

For algorithmic purposes, an *array* or *list* data structure can be considered equivalents here:

|  |  |
| --- | --- |
| Algorithm 1 | Algorithm 2 |
| **BEGIN**  **PRINT** "USE UPPERCASE, NO SPACES"  plain\_text = input from user  num\_chars = *length*(plain\_text)  plain\_chars = *array*(plain\_text)  i = 0  cipher\_chars = empty array    **WHILE** i < num\_chars  temp\_char = plain\_chars[i]  temp\_ord = *ord*(temp\_char)  norm\_ord = temp\_ord – *ord*("A")    shift = (norm\_ord + 3) *modulo* 26  new\_ord = shift + *ord*("A")  new\_char = *chr*(new\_ord)  cipher\_chars[i] = new\_char  char\_index = char\_index + 1  **ENDWHILE**  **PRINT** *str*(cipher\_chars)  **END** | **BEGIN**  num\_chars = 0  **WHILE** num\_chars < 1  **PRINT** "USE UPPERCASE, NO SPACES"  plain\_text = input from user  num\_chars = *length*(plain\_text)  **ENDWHILE**  plain\_chars = *array*(plain\_text)  i = 0  cipher\_chars = empty array  alphabet = *array*("A","B", .. "Z")    **WHILE** i < num\_chars  temp\_char = plain\_chars[i]  **IF** temp\_char *in* alphabet **THEN**  temp\_ord = *ord*(temp\_char)  new\_ord = temp\_ord + 3  new\_char = *chr*(new\_ord)  cipher\_chars[i] = new\_char  **ENDIF**  char\_index = char\_index + 1  **ENDWHILE**  **PRINT** *str*(cipher\_chars)  **END** |

Evaluate the two algorithms to identify which statement is the most correct.

1. Algorithm 1 is more accurate and provides a greater level of safety than Algorithm 2.
2. Algorithm 1 is more accurate but provides less safety than Algorithm 2.
3. Algorithm 1 is less accurate but provides a greater level of safety than Algorithm 2.
4. Algorithm 1 is less accurate and provides less safety than Algorithm 2.

# Section 2

## Instructions

* Write using black or blue pen.
* Respond in paragraphs consisting of full sentences.
* If you need more space for a response, use the additional pages at the back of this book.
  + On the additional pages, write the question number you are responding to.
  + Cancel any incorrect response by ruling a single diagonal line through your work.
  + Write the page number of your alternative/additional response, i.e. See page …
  + If you do not do this, your original response will be marked.
* This section has five questions and is worth 60 marks.

**Question 11 (16 marks)**

1. Explain how *Quality of Service* can be impacted by latency and jitter when transmitting data packets. [*4 marks*]
2. Explain the functional difference between TCP and IP [*4 marks*]
3. Explain how TCP guarantees packet delivery [*4 marks*]
4. Explain the difference in use or application of HTTP and FTP [*4 marks*]

**Question 12 (8 marks)**

Analyse the following JSON and XML examples.

|  |  |
| --- | --- |
| JSON | XML |
| { "catering": {  "year8": {  "sausages": "heaps",  "sides": ["salad", "sauce"]  },  "year9": null,  "year10": { "cookies": 170 }  }  } | <jobs>  <job category="fastfood">  <location>KFC</location>  <conditions>part time</conditions>  <payperhour>21</payperhour>  </job>  <job category="retail">  <location>IKEA</location>  <conditions>casual</conditions>  <payperhour>19</payperhour>  </job>  </jobs> |

1. Annotate the JSON example above to identify only one of each of the following: [*4 marks*]
   1. Key
   2. Value
   3. Array data type
   4. Key / value pairing separator
2. Annotate the XML example above to identify only one of each of the following: [*4 marks*]
   1. Root element
   2. Element
   3. Text
   4. Attribute

**Question 13 (16 marks)**

An ID barcode reader for a school library uses a scanner to input and process characters one at a time. Valid ID barcodes must:

* Begin with the character 'S' for student or 'T' for teacher, followed by a sequence of integers
* The sequence of integers must be 5 in length, therefore a valid ID barcode will contain 6 characters in total (including the 'S' or 'T'). For example, S12345 or T98765 are both valid
* If the barcode is invalid, the algorithmic module returns an *empty array* (list). Otherwise, it returns an array / list containing the valid character sequence of the barcode.

The algorithm shown in Stimulus 1 was designed to check if a given barcode is valid. Examine it, then answer the questions that follow.

|  |  |
| --- | --- |
| Stimulus 1: Barcode scanner algorithm | |
| 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25  26  27 | **BEGIN**  **SET** Barcode = [***INPUT*** *barcode string into array]*  **SET** Result = [] //empty array that will allow 6 characters  **SET** Processed = False  **SET** Valid = True  **SET** Counter = 0  **WHILE** **NOT** Processed  **SET** Character = Barcode[Counter]  **IF NOT** ***digit***(Character) **OR** Counter == 0 **THEN**  **IF** Character == 'T' **THEN**  **SET** Result[Counter] = 'T'  **ELSE**  **SET** Result[Counter] = 'S'  **SET** Valid = True  **ENDIF**  **ELSE**  **SET** Barcode[Counter] = Character  **SET** Valid = False  **IF** Counter == 5 **THEN**  **SET** Processed = True  **SET** Valid = True  **ENDIF**  **ENDIF**  **SET** Counter = Counter + 1  //trace values here for question (a)  **ENDWHILE**  **RETURN** Barcode  **END** |

|  |
| --- |
| **Stimulus 1 in Python:**  [**https://digisoln.com/skulpt/python/demo/barcode\_scanner\_algorithm\_q13\_mock.py**](https://digisoln.com/skulpt/python/demo/barcode_scanner_algorithm_q13_mock.py) |

***As this is a practise, you may wish to complete this question digitally using the above link.***

1. Using the table below, desk check the algorithm using the barcode T12X45 as input, tracing the values of the variables Character, Result and Valid at Line 24.

There may be errors such as *infinite loop* or *program crash* in the current state of algorithm. If you experience any of these errors, then exit the trace. You may not require all of the space given in the table below. The first value of Character has been pre-completed for you: [*4 marks*]

|  |  |  |
| --- | --- | --- |
| Character (string) | Result (array values) | Valid (Boolean) |
| T |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

1. Explain any *errors*, *anomalies,* or *inefficiencies* you encountered during the desk checking process in the previous question. [*4 marks*]
2. Refine the algorithm to correct the *errors* or *anomalies,* or *inefficiencies* you discovered in the previous question [*8 marks*]

**Question 14 (8 marks)**

Stimulus 2 below is an extract from a juice company database. It shows ingredients, pricing, orders, as well as the current status of these orders.

|  |
| --- |
| Stimulus 2: juice company database |
| |  |  |  | | --- | --- | --- | | **Recipes** | | | | juice | ingredient | grams | | CreativityCure | Chia | 200 | | CreativityCure | Quinoa | 200 | | CreativityCure | Spirulina | 200 | | JazzyJuice | Acai | 300 | | JazzyJuice | Kale | 150 | | JazzyJuice | Spirulina | 150 | | ReplenishingRefresher | Blueberries | 75 | | ReplenishingRefresher | Chia | 75 | | ReplenishingRefresher | Kale | 300 | | ReplenishingRefresher | Quinoa | 150 |  |  |  | | --- | --- | | **Updates** | | | orderNum | status | | | 1 | complete | | | 2 | complete | | | 3 | complete | | | 4 | delivering | | | 5 | delivering | | | 6 | mixing | |      |  |  |  | | --- | --- | --- | | **Pricing** | | | | juice | retailPrice | costPrice | | CreativityCure | 7.5 | 2.25 | | JazzyJuice | 7.5 | 1.5 | | ReplenishingRefresher | 10 | 3 |  |  |  |  | | --- | --- | --- | | **Orders** | | | | orderNum | juice | quantity | | 1 | CreativityCure | 2 | | 1 | JazzyJuice | 3 | | 1 | ReplenishingRefresher | 1 | | 2 | CreativityCure | 1 | | 2 | JazzyJuice | 1 | | 2 | ReplenishingRefresher | 2 | | 3 | ReplenishingRefresher | 1 | | 4 | ReplenishingRefresher | 2 | | 5 | JazzyJuice | 1 | | 6 | JazzyJuice | 1 | | 6 | ReplenishingRefresher | 1 | |

The following query was developed to show the business *profit* (meaning the total cost of all drinks sold at retail price) from completed orders, although it gives an unexpected result:

**SELECT** SUM(retailPrice \* quantity) **AS** Profit

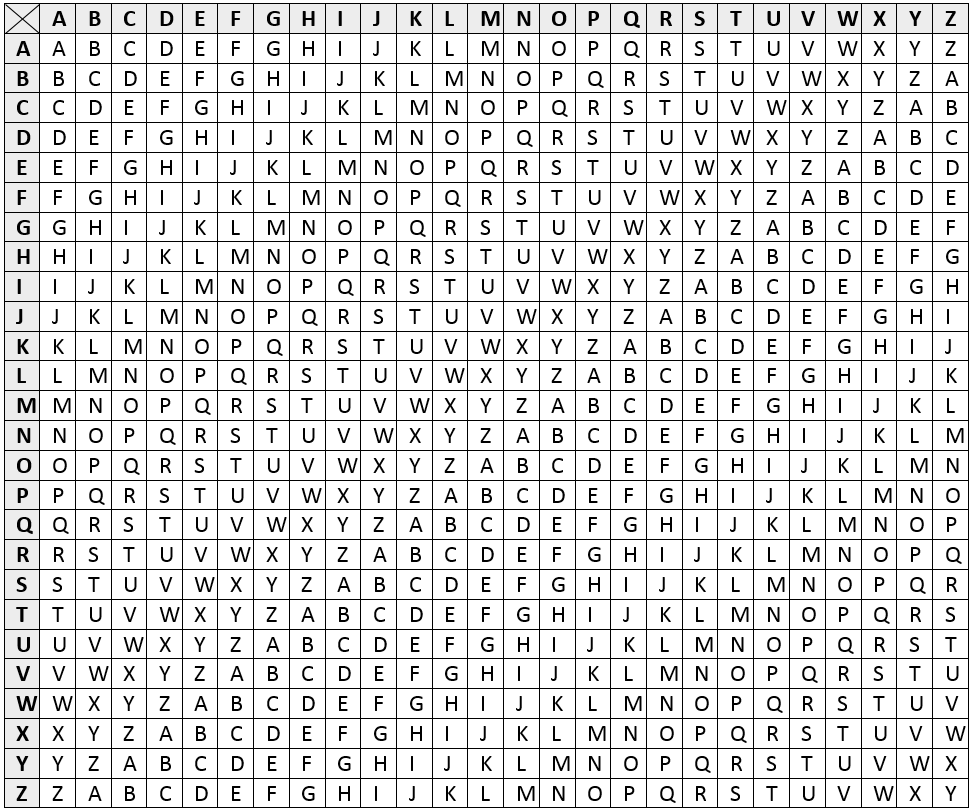
**FROM** Pricing JOIN Orders JOIN Updates

**WHERE** status = "complete"

1. Refine the query so that it works as intended. [*4 marks*]
2. Juices that use Chia seed have been recalled. Refine the query in question 14(a) so that it removes juices sold that contain Chia seed. [*4 marks*]

**Question 15 (12 marks)**

1. Find the cipher text result of encrypting the plain text word “HELLO” with the key “XYZ” using the following Vigenere square [*2 marks*]:



1. Develop an algorithm using pseudocode to demonstrate the encryption process used in question A. [*10 marks*]

# Section 3

## Instructions

* Write using black or blue pen.
* Respond in paragraphs consisting of full sentences.
* This section has one question and is worth 30 marks.

**Question 16 (30 marks)**

Stimulus 3: The Luhn algorithm, also known as the modulus 10 or mod 10 algorithm, is a simple checksum formula used to validate a variety of identification numbers, such as credit card or IMEI (International Mobile Equipment Identity) numbers. The algorithm works as follows:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Step 1: Input a number (for example 918243) and separate the digits: | | | | | |
| 9 | 1 | 8 | 2 | 4 | 3 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Step 2: Reverse the number | | | | | |
| 3 | 4 | 2 | 8 | 1 | 9 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Step 3: Double every second number | | | | | |
| 3 | 8 | 2 | 16 | 1 | 18 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Step 4: If a result from step 3 was a two digit number, then add those two digits together: | | | | | |
| 3 | 8 | 2 | 1 + 6 = 7 | 1 | 1 + 8 = 9 |

|  |  |
| --- | --- |
| Step 5: Add up all the digits | |
| 3 + 8 + 2 + 7 + 1 + 9 = | 30 |

|  |  |
| --- | --- |
| Step 6: If the result from step 5 **modulo 10 = 0**, then the number is valid. Otherwise, it is invalid. | |
| 30 mod (%) 10 = 0 … | VALID |

Other examples of valid numbers include:

* 893
* 2428
* 00117
* 00

Examples of invalid numbers include:

* 1234
* 99
* 0

1. Define pseudocode to implement the Luhn algorithm check on any user input as it has been illustrated in Stimulus 3 [*10 marks*]
2. The Stimulus 3: Luhn algorithm is one example of ensuring an online transaction is safely and securely completed.

An online transaction in this context can be defined as the purchase of goods or services through a website by a customer.

Describe three other measures that could be implemented during the transfer or storage of data involved within an online transaction to ensure it is kept *safe and secure*, and evaluate the comparative effectiveness of each measure. [*10 marks*]

1. Use a level 1 data flow diagram to symbolise the information flow through the following online shopping system context [*10 marks*]

|  |
| --- |
| *Drop shipping* is a method of online retailing where a vendor (known as a seller) can sell a product to a customer directly from a supplier, and once sold, the supplier fulfills the delivery of the product directly to the customer. This way, the vendor does not have to purchase any initial stock from the supplier in order to commence selling, and the supplier can save money from not having to spend on creating or maintaining retail or online outlets.  An online vendor by the name of “Party Platters” has curated a list of products sourced from two different suppliers, namely a local butcher “Marks Meats” and a local delicatessen “Charlies Cheeses”. She is *drop shipping* these products via her Party Platters online shopping system. Each product sourced by Party Platters from its suppliers contains a unique product identification number, label (name), category, description and URL image link.  The online shopping system allows users to browse, filter by substring search or category, view descriptions and images, and purchase products by adding them to a “cart”. The cart is managed by a server-side session variable, and when the purchase is complete and validated, the order is forwarded to the suppliers, who are responsible for delivering the products to the customer.  The validation process occurs at checkout, where the customers payment details (either credit card or PayPal account) are validated by an embedded PayPal merchant services script. The script is called remotely on PayPal servers, with the customers payment details supplied as parameters to the functions contained within the script. The script returns a valid or invalid transaction result.  Additionally, the online shopping system has a feedback system, which can be completed by a customer after a purchase is complete to rate their experience. The online shopping system stores customer data after a successful purchase, including payment details, in order to expediate purchases in returning / repeat business. The online shopping system also records a full transaction history for each customer, in order to analyse popular products or to better inform other business decision making. |