|  |
| --- |
| Robert van der Spoel |
| IT Graad 12 PAT – Fase 1 |
| **Volitant Express** |
| **A logo of an airplane** |

# **TASK 1A : SCENARIO / SCOPE**

## **Topic:** Volitant Express- A program that allows companies to place an order to fly items from one country to another and manages the freight

## **Purpose of program:** More Companies are transporting items globally and there has been an increase in time sensitive deliveries, thus a program is needed that can manage and control this process and make it easy and efficient for users to use.

## **Possible solution:** The program allows a company to easily place an order to transport an item from one country to another and then calculates the cost, time and distance, it also allocates a plane ,perfect for the job, to get the item to the desired destination in a short amount of time. It plans the freight of the selected item for the client company

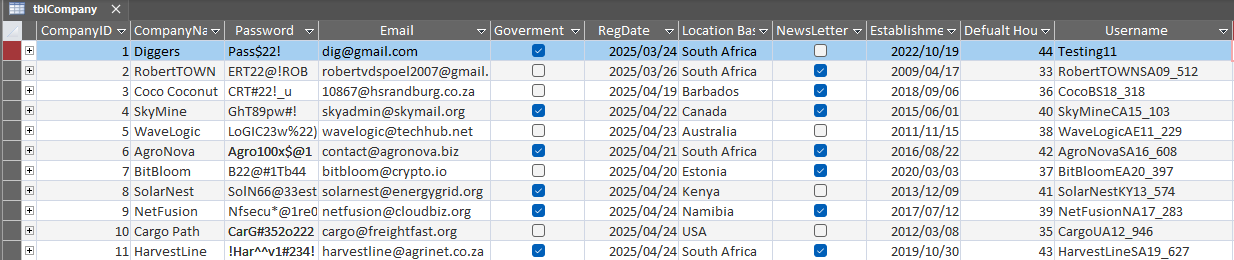
## **Scope:** My program cannot accept actual payments as it only simulates the transactions. My program cannot accept/ process orders with more than one item in the order. My program cannot validate a real email and accepts that the email is real if the format is correct. My program cannot encrypt companies data to keep it secure from hackers. Some cost calculations and flight data may not be fully accurate therefore total cost calculations and durations may be inaccurate. My program can’t be accessed by multiple users at once.

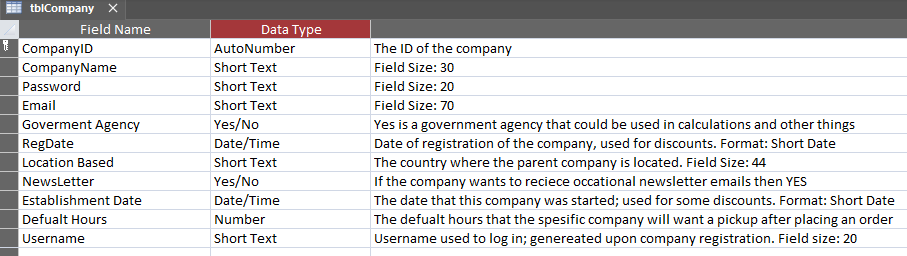
# **TASK 1B : USER REQUIREMENTS**

| **WHO (Role)** | **WHAT (Activity)** | **LIMITATION** |
| --- | --- | --- |
| Clients (Companies)  Placing orders to transport items between 2 countries | They can Register, Login, Change their companies details, receive newsletter emails, place orders for items to be transported between countries, pay online for the order that they have placed, view past payment of orders and unpaid orders, view a log of their history of placing orders at Volitant express | They cannot add new items or categories for shipping, they cannot see summaries of all the data in the system, they can’t send emails, they can’t see details regarding other companies orders and they cannot see the details of other companies, cannot type custom SQL statements |
| Admin  Manage and control the system | They can add a new plane, update info about planes, add a new item and manage existing items, see detailed info about orders, see companies details, send newsletter emails and see system summaries, change the status of an order | They cannot type custom SQL statements, cannot place orders or manage an order as if they were a company customer |
| Admin Developer  Admin with an extra feature access | They can add a new plane, update info about planes, add a new item and manage existing items, see detailed info about orders, see companies details, send newsletter emails and see system summaries, change the status of an order, they can also type custom SQL statements | They cannot place orders or manage an order as if they were a company customer |

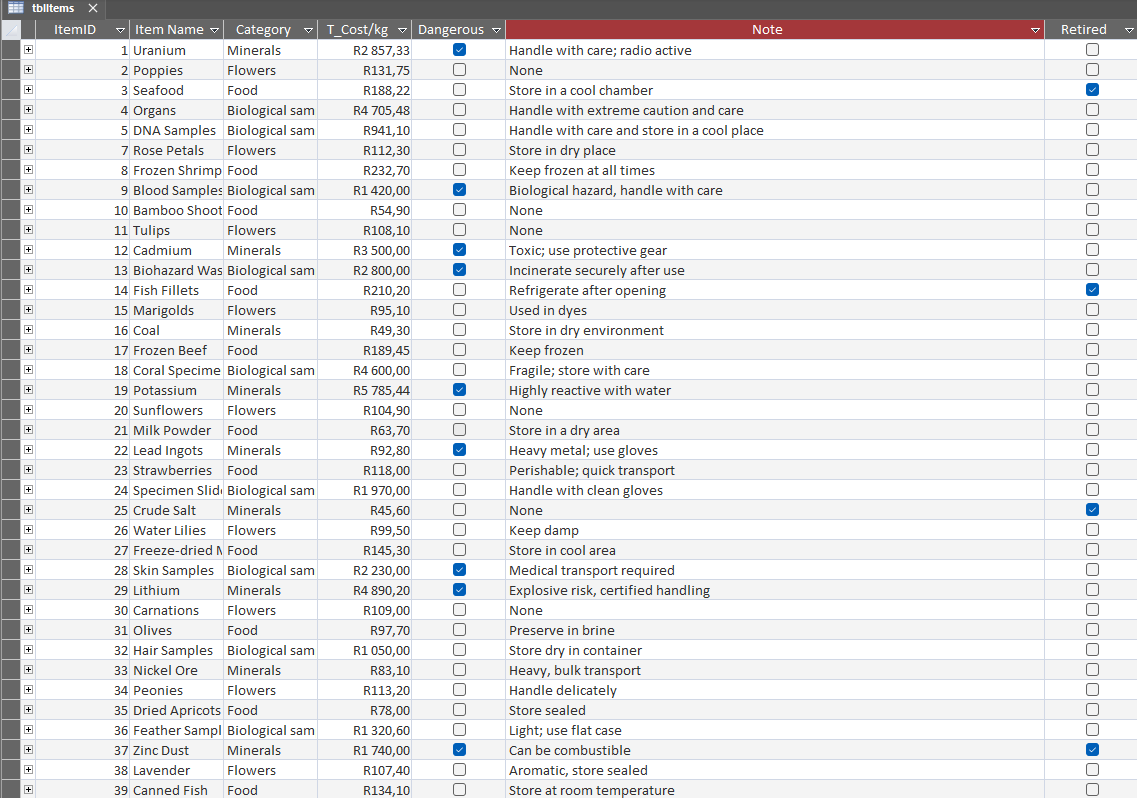
# **TASK 2 : DATABASE DESIGN**

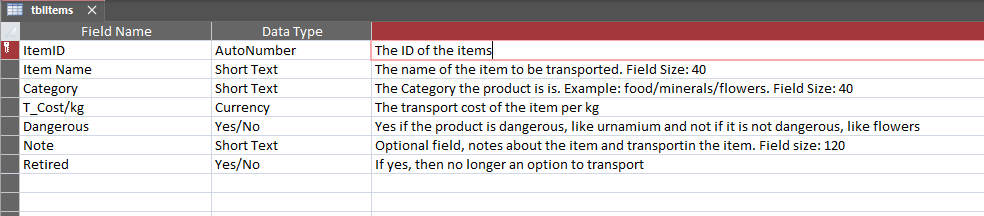
tblCompany:



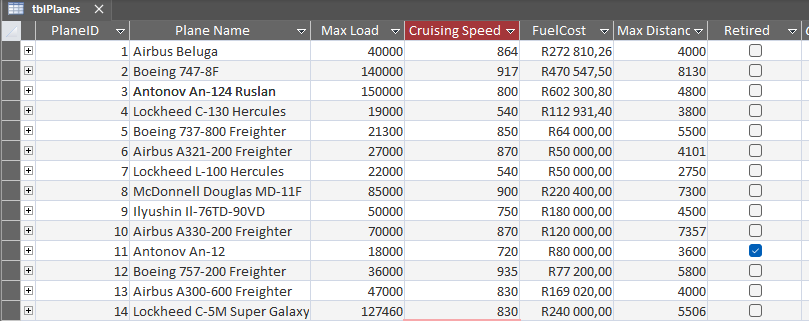


tblItems:





tblPlanes:



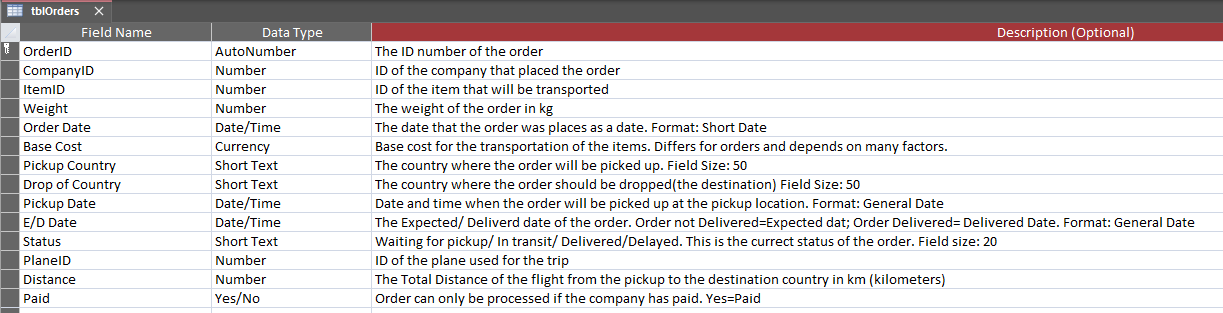
A screenshot of a computer

AI-generated content may be incorrect.

tblOrders:

A screenshot of a computer

AI-generated content may be incorrect.



Relationship:

A screenshot of a computer

AI-generated content may be incorrect.

# **TASK 3A : CLASS DESCRIPTION AND CLASS DIAGRAM**

|  |
| --- |
| Tdistance |
| Tdistance will receive the Coordinates of the two countries and also the drop off and pickup counties names. The distance will the be calculated between these 2 countries using the Haversine formula. Afterwords the Tostring will be created that will contain info regarding the trip like coordinates on earth, countries and the distance of the flight. The distance is used in regards to planning the trip, costs and plane |
| Attributes |
| -fLatitudeStart, fLongitudeEnd, fLatitudeStart, fLongitudeEnd, fDistance : real;  -fPickupCountry, fDropOfCountry: string; |
| Methods |
| + Constructor Create(pLat1, pLat2, pLong1, pLong2: real; fPickupCountry, fDropOfCountry: string);  -CalculateDistance : real;  - RealCoordinates: string;  + ToString : string;  + GetDisrance : real; |

# **TASK 3B : TEXT FILES(s)**

|  |  |
| --- | --- |
| **Name of text file** | Country\_Coordinates.txt |
| **Purpose** | To store the coordinates of every country in the world |
| **Format** | CountryCode#Latitude#Longitude#Name |
| **Extract of data** | ZA#-30.559482#22.937506#South Africa |

# **TASK 3C : ARRAYS**

Latitude = North/ South of the equator

Longitude = East/ West of Greenwich

|  |  |
| --- | --- |
| **Declaration** | arrCountryName: array[1..245] of string;  arrCountryCode : array[1..245] of string;  arrLatitude: array[1..245] of real;  arrLongitude : array[1..245] of real; |
| **Populate arrays** | Populate from Country\_Coordinates.txt text file  Stores the coordinates of all the countries that is stored in the applicable txt file |
| **Processing** | -Sort the arrays alphabetically from A to Z using their names  -Search for countries that contain a part of a word in their name or search using their country codes  -Convert the latitude/ longitude into radians when reading into the array for calculations use  -Find countries in a latitude/ longitude range  -Use country code to load country flag when registering |

# **TASK 4A : NAVIGATION / FLOW DIAGRAM**

See the next page please

A diagram of a flowchart

AI-generated content may be incorrect.

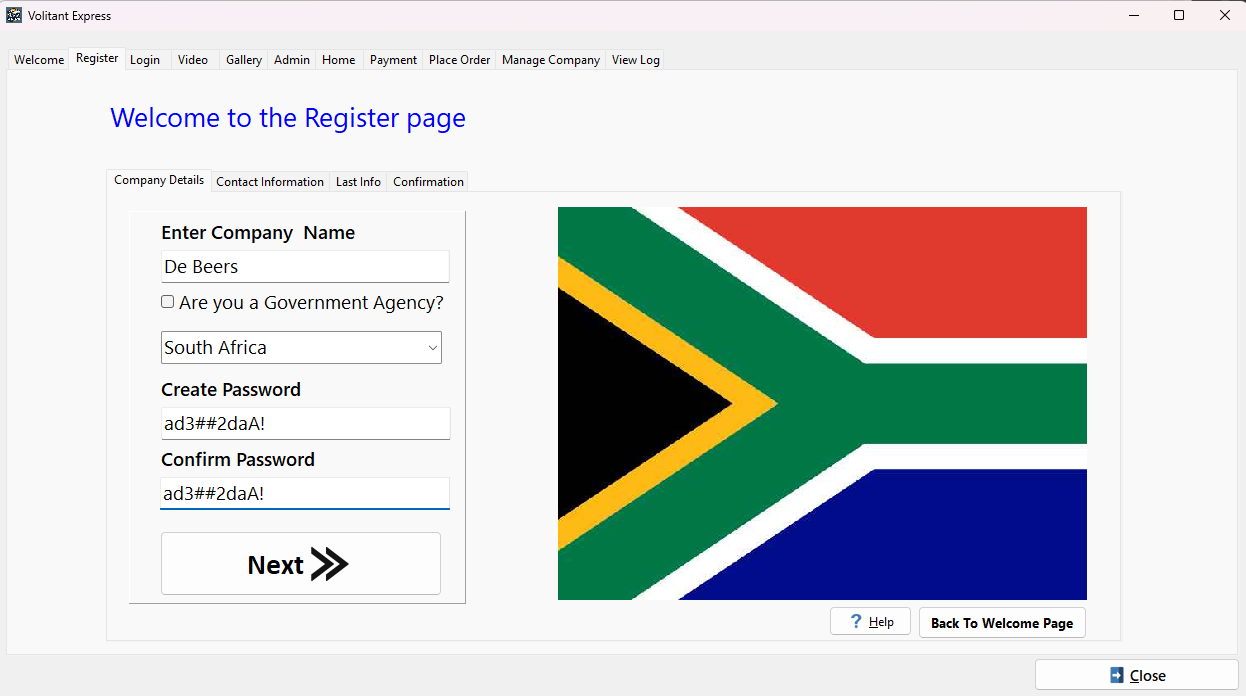
# **TASK 4B : GUI DESIGN**

tsWelcome

A screenshot of a computer

AI-generated content may be incorrect.

tsRegister



A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

tsLogin

A screenshot of a login page

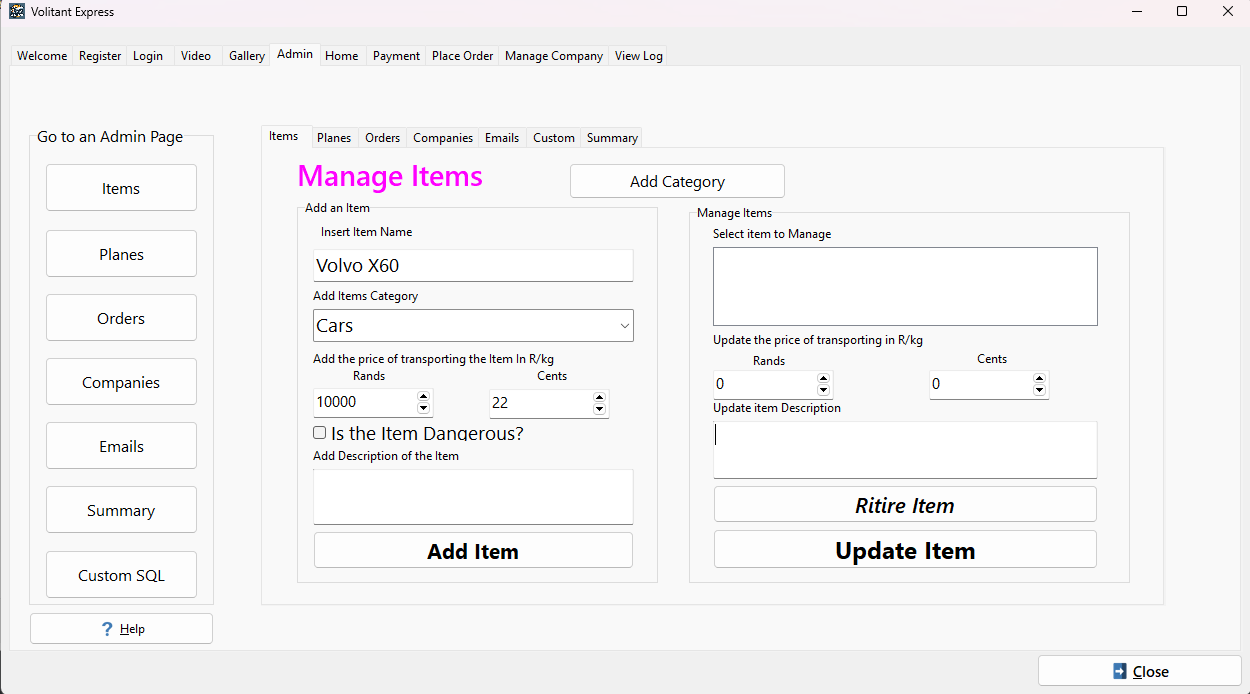
AI-generated content may be incorrect.

tsIntroVideo

A person in a suit and tie

AI-generated content may be incorrect.

tsAdmin



A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a email

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

tsHome

**A screenshot of a computer

AI-generated content may be incorrect.**

tsPayment

A screenshot of a computer

AI-generated content may be incorrect.

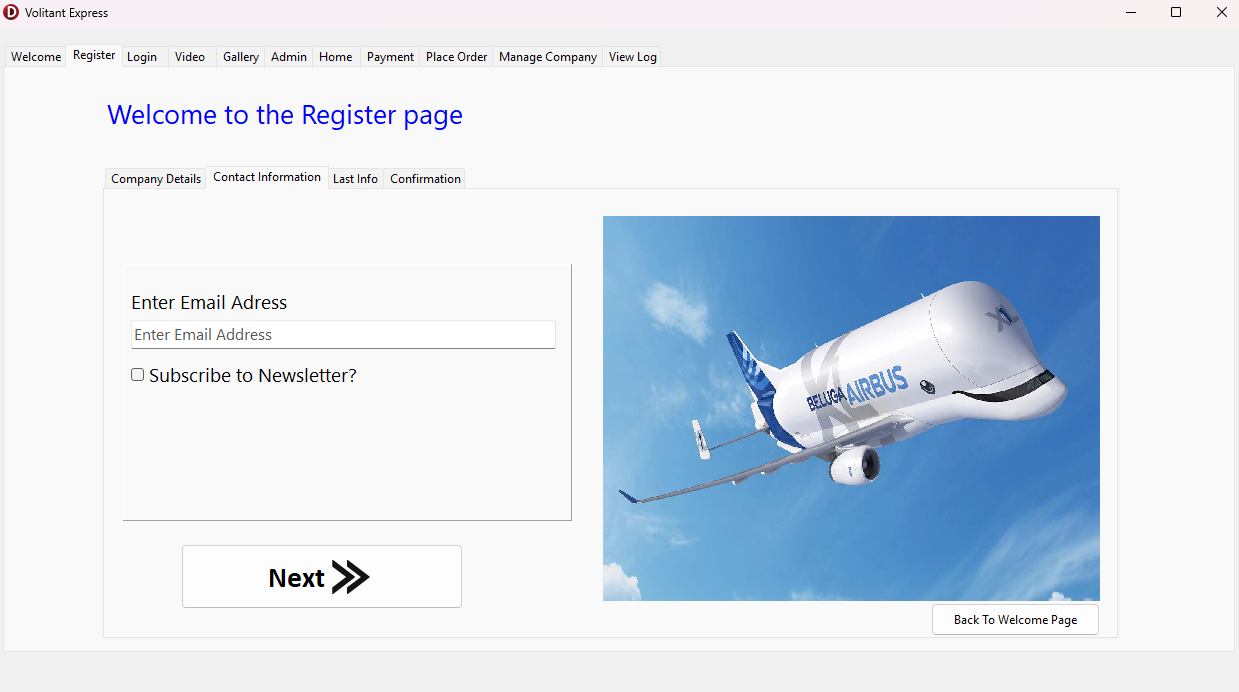
# **TASK 5 : IPO – SOFTWARE DESIGN TOOL**

## **TASK 5 : INPUT:**

Tabsheet 1: tsRegister

A screenshot of a computer

AI-generated content may be incorrect.



A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

\*Most of the components on the last page is for output use, only the checkbox is for input use

| **Source** | **Data Type** | **Format (size, M/F for gender, yyyy/mm/dd)** | **GUI Component** |
| --- | --- | --- | --- |
| Keyboard | sCName: string | Store as a string, less than 30 characters. This is the persons name | edtCName |
| Mouse Click | bGAgency : Boolean | If checked, true, then the company is a Government agency, else (False) the company is not a government agency | chkGovernment |
| Keyboard | sPasswordCreate : string | Store as a string, shorter than 20 characters and longer than 6 | edtCreatePassword |
| Keyboard | sPasswordConfirm: string | Match the password entered in the first password edit | edtConfirmPass |
| Keyboard | sEmailReg : string | Valid email address; validated/  Must include an @  Must not be longer than 70 characters | edtRegEmail |
| Mouse click | bNewsLetter: Boolean | If checked (True) then the company will be subscribed to the Newsletter, if left unchecked, the user will not be subscribed | chkNewsLetter |
| Mouse/Keybaord | dDateOfEstablishment; Tdate | Yyyy/mm/dd on my home PC  \*Could differ on another PC, however this will not effect the program  Store as a date variable that the user selects from the date time picker | dptEstablishedDate |
| Mouse/Keybaord | iDefualtHours; integer | Stored in hours  Integer for the default hours to use from when an order is placed and when it is to be picked up  A number of hours between 24 and 300 | sedRegDefaultHours |
| Mosue | bConfirmTrue, Boolean | If the checkbox is selected = True; then the user confirms that the info that they provided is true and accurate | chkConfirmRegInfo |

Tabsheet 2: tsAdmin

A screenshot of a computer

AI-generated content may be incorrect.

| **Source** | **Data Type** | **Format (size, M/F for gender, yyyy/mm/dd)** | **GUI Component** |
| --- | --- | --- | --- |
| Keyboard | sAddItemName : string | Store as a plane string; less than 40 characters | edtItemAddName |
| Mouse/ keyboard | sItemCatergory: string | Store as a string, will be limited to what is in the combo box. A category must be less than 40 characters | cmbItemCategoryAdd |
| Mouse/ keyboard | iItemRands: integer | The rands of the items price, more that 1 and less than 15000 | sedAdditemRands |
| Mouse/Keyboard | iItemCents: integer | The cents of the items price, should be between 0 and 99 | sedAdditemCents |
| Mouse | bDangerous: Boolean | If the item is dangerous, like radioactive items, them select so that it is True | chkDangerousItemAdd |
| Keybaord | sAddDescription | A description of the item being added, includes transport notes and other things. Optional. Should be less than 120 characters | redAddDescription |
| Mouse/ keyboard | iItemUpdateRands: integer | The rands of the items price, more that 1 and less than 15000 | sedUpdateItemRands |
| Mouse/Keyboard | iItemUpdateCents: integer | The cents of the items price, should be between 0 and 99 | sedUpdateItemCents |
| Keybaord | sUpdateItemDescription: string | An update to the description and notes of an item. Stored as a string with a length of 120 characters or shorter | redUpdateItem |
|  |  |  |  |

## **TASK 5 : VALIDATION:**

| **Input validated** | **Validation type and description** | **Associated error message** |
| --- | --- | --- |
| **Register** |  |  |
| Password | Format, Range check   * Longer than 6 characters * Shorter or equal to that 20 characters   (Use length function in Delphi)   * Should contain a number * Should contain a special characters * Should contain a capital Letter   (Use IN function in Delphi to check that there is one of each present in the password inputted string)  Presence Check:  \*Here I am also checking for NULL as if nothing was entered, none of the conditions would be met | Enter a valid password with a length of 6-20 characters; containing a number, a special character and a capital letter. |
| Company Name | Presence Validation  Name Should not be empty; name not equal to an empty string | Enter a name |
| Company Name | Range Check; Length, should be larger than 1 and smaller or equal to 30 characters | Name Length should be between 1 and 30 characters |
| Company Name | Format Check; Last Character may not be a space | The last character of the name may not be a space |
| Company Name | Format Check; Check that the entered name only contains alphabet characters and certain special characters such as a . or - using IN[] | Only alphabet characters and certain characters are allowed in the name |
| Country Based | Presence Check; Test if a value was selected in the selection component use .itemindex | Please select a country from the select country based combobox |
| Confirmed Password | Format Check; Must match the Create Password edit exactly | Passwords do not match |
| Email Address | Presence Check; Check that an email address wat entered | Please enter an email address |
| Email | Format Check; Check that the email address is valid:   * Must contain 1 @ * Must not have any spaces * Must have a **.**Something * Must be 70 characters or shorter in length (Range Check) | Please enter a valid email address:  Your email address:   * Must contain 1 @ * Must not have any spaces * Must have a **.**Something * Must be 70 characters or shorter in length |
| Date of establishment | Range Check; Check that the Companies date of establishment is not in the future | Please enter/select a valid date of establishment |
| All of the entered data is true and valid | presence Check; A confirmation in the form of a checkbox and Boolean where the checkbox must be selected and therefor True to allow the company to complete their registration | You must confirm that the information that you entered in the registration process is true and accurate |

|  |  |  |
| --- | --- | --- |
| **Admin** |  |  |
| Item name | Format Check; Should only contain alphabet characters and spaces | Please enter a valid item name |
| Item Category | Presence Check; Test that an item category has been selected | Please select the category that the item falls into |
| Price for transportation | Range Check; Check that it is a value larger than R0 and smaller than R15000 | Enter a valid cost for transport; between R 1 and R 15000 |
| Description of item | Range Check; Check that the description if item is not longer than 120 characters | Item description should be less than 120 characters.  Current: \_\_ characters |
| Item to manage | presence Check; An item must be selected to manage that item, itemindex must be larger than -1 | Please select an item to manage |
| Updated item price | Range Check; Check that it is a value larger than R0 and smaller than R15000 | Enter a valid cost for transport; between R 0 and R 15000 |
| Updated  Description of item | Check that the description if item is not longer than 120 characters | Item description should be less than 120 characters.  Current: \_\_ characters |
| Text file containing categories | A type of presence check (existence check); Check to see if the txt file that contains the item categories exists; text file data type | Item\_Categories.txt not found.  A new file to replace it has been created. Add new categories to this file |

## **TASK 5 : PROCESSING**:

|  |  |
| --- | --- |
| **What processing will be done** | **Algorithms, formulas, etc.** |
| **List:** | **Algorithms: (How will the processing be done)** |
| 1 Match (Choose) a plane for a specific order |  |
| 2 Send the newsletter |  |
| 3 Choose and load images to the gallery |  |
| 4 Log Into the account |  |
| 5 Calculate the total Cost of an order |  |
| 6 Create a Username for the Company | Algorithm 1  Company Name  ← input from user  Country Based ← Input from user  Year Established ← Input from user as the date that the company was established  Random Number ←Randomly generated 3 digit number  Pos← Pos(' ', Company Name)   // (Position Of ‘ ’ in Company Name)  NameUse←Company Name  If Pos = 0 then  Begin      if Length(Company Name) > 14 then      begin      NameUse← Copy(NameUse, 1, 14);      end;  End  else  begin  NameUse← Copy(Nameuse, 1, Pos-1)  end;  CountryBased←Uppercase( CountryBased[1] +  CountryBased[Length(CountryBased)])  Year\_Establihed\_int ← YearOf(Year Established )  YearEstablished\_String ← InttoStr(Year\_Establihed\_int[3]) +   Inttostr(Year\_Establihed\_int [4])  Username ← Company Name + CountryBased  + YearEstablised\_String +'\_'+ random Number | |
| 7 Check that there are no invalid characters in the password | Algorithm 2  sRegisterPassword← Input from user  arrAllowedCharacters← Array populated at startup; contains allowed characters  Loop for cChar IN sRegisterPAssword  begin    bErrorCharacter←True  Loop for c in arrAllowedCharacters  begin  if (cChar = c) then  bErrorCharacter ← False;  Break ; // Not from curriculem, will however, improve performance  end;  if bErrorCharacter = True then  begin  ShowMessage('Invalid/Unsupported character in Password') ;  exit;  end;    end; |
| 8 Sort the countries according to the alphabet from A to Z | Algorithm 3  iCount←245  arrCountryName← input from txt file  arrCountryCode← input from txt file  arrLatitude← input from txt file  arrLongitude← input from txt file  iKeep←0  sKeep←''  Loop for iOut from 1 to (iCount-1)  begin  Loop for iIn from iOut+1 to Icount  begin  if Uppercase(arrCountryName[iIn]) < Uppercase(arrCountryName[iOut]) then  begin  sKeep← arrCountryName[iOut];  arrCountryName[iOut]← arrCountryName[iIn];  arrCountryName[iIn]← sKeep  sKeep← arrCountryCode[iOut];  arrCountryCode[iOut]← arrCountryCode[iIn];  arrCountryCode[iIn]← sKeep  iKeep← arrLatitude[iOut];  arrLatitude[iOut]← arrLatitude[iIn];  arrLatitude[iIn]← iKeep;  iKeep← arrLongitude[iOut];  arrLongitude[iOut]← arrLongitude[iIn];  arrLongitude[iIn]← iKeep;  end;  end;  end; |
| 9 List all the orders of each company | Algorithm 4  sItemName←''  tblCompany.First  while not end of tblCompany  begin  tblOrders.First  richedit.lines.add(#10+tblCompany['CompanyName'] + #9+ tblCompany['Location Based']+ #10)  while not end of tblOrders  begin  if tblCompany['CompanyID'] = tblOrders['CompanyID'] then  begin    tblItems.First;  bFound←False;  while not (end of tblItems) and (bFound = False) Do  begin  if tblOrders['ItemID'] = tblItems['ItemID'] then  begin  bFound←True;  sItemName← tblItems['Item Name']  richedit.lines.add(tblOrders['Weight'] +#9 + tblOrders['Drop of Country']+ #9 + tblOrders['Pickup Country'] + sItemName)  end  else  tblItems.Next;  end;    sItemName←'';  tblOrders.Next;  end;  tblCompany.Next  end;  end; |

## **TASK 5 : OUTPUT:**

## Tabsheet 1: tsRegister

A screenshot of a computer

AI-generated content may be incorrect.

| **Data to Output** | **Format (Type, size)** | **Output Component** |
| --- | --- | --- |
| Username | Display the generated username of the user in the Colour red. Display as a string data type.  Format: Name of Company (First word of name) and cutoff on length of 14 characters + First character of Country Based as a capital + last character of Country Based as a capital + last two digits of year Established + 3 digit generated number  The Total length will have to be 21 characters or shorter | edtGUsername |
| Password | Display the Password; use Navy colour. Display as a string  Your Password: \_\_\_\_\_\_\_\_\_ | pnlConfirmPass |
| Company Age | Calculate the company’s age today using their establishment date | sedCompanyAgeConfirm |
| Government Agency | Yes/No selected  Display as a string  Government Agency: YES/NO | redConfirmRegInfo |
| Country Based | Display as a string  Country Based: \_\_\_\_\_\_\_\_\_ | redConfirmRegInfo |
| Email | Display as a string  Email: \_\_\_\_\_\_\_\_@\_\_\_ | redConfirmRegInfo |
| Newsletter | Yes/No selected  Display as a str  Newsletter: YES/NO | redConfirmRegInfo |
| Default Hours | Display as a integer + a string  Default hours: \_\_ h | redConfirmRegInfo |

Tabsheet 2: tsAdmin

A screenshot of a computer

AI-generated content may be incorrect.

| **Data Output** | **Format (Type, size)** | **Output Component** |
| --- | --- | --- |
| A tables contents | Displayed according to the format of the table, as the data is stored | dbgDifferentTables |
| Company Name, Item name, Plane name | When clicking on a record from tblOrders, display all the foreign data that relates to that record underneath each other. Display as a string:  Company Name: \_\_\_\_\_  Item Name: \_\_\_\_\_  Plane Name: \_\_\_\_\_ | Showmessage dialogue |
| Total Accounts counts | The total accounts of companies as a string and integer  Format:  Total Accounts: \_\_\_ | pnlTotalAccounts |
| Total Orders of the System count | The total amount of orders placed as a string and integer  Format:  Total Orders: \_\_\_ | pnlTotalOrders |
| Total Items that can be transported | The total items displayed as a string and an integer  Format:  Total Items: \_\_\_\_ | pmlTotalItems |
| The total active plane types that Volitant Express uses | Only plane types that are NOT retired will be counted. Display as a string and an integer  Format:  Total Active Plane Types: \_\_\_\_ | pnlTotalActivePlanes |
| The most used plane model | Output as a string  Format:  Top Plain: \_\_\_\_\_\_ | lblTopPlain |
| The most popular item that is shipped | Output as a string  Format:  Top Item: \_\_\_\_\_\_ | lblTopItem |
| The most used Pickup Country | Output as a string  Format:  Top Pickup Country: \_\_\_\_\_\_\_ | lblTopPickupC |
| The most accessed drop-off country | Output as a string:  Format:  Top Drop-Off Country: \_\_\_\_\_\_\_ | lblTopDropOffC |
| The total revenue that Volitant Express has made | Output as a real; that is currency formatted to 2 decimal places  Format:  Total revenue: \_\_\_\_\_\_ | lblTotalRevenue |
| The average weight of an order in kg | Displayed as a real with 2 decimals  Format:  Average order weight: \_\_\_\_\_\_\_ kg | lblAverageWeight |
| Volitant Expresses progress to reaching their revenue goal | Progress displayed in a progress bar that is green. It calculates how far the company has come with their progress and plots it on the bar | PBrevenue |
| Volitant Expresses progress to reaching their total orders goal | Progress displayed in a progress bar that is green. It calculates how far the company has come with their progress and plots it on the bar | PBorders |