|  |  |
| --- | --- |
| **IT Graad 11 PAT 2024 : Healthy Living** | |
| **Naam en Van:** | **Robert van der Spoel** |
| **Projek Naam:** | **The Gym Membership** |

# **Task 1A: Define the task**

## Topic: A program that allocates points to users who workout and gym based on how much effort they put into their exercise and how often they exercise. It is for any person who wants to workout, however, specifically for people who require a bit of motivation for working out and want their workouts to deliver more than a workout; to get rewarded for working out.

## Purpose of program: People are living in an unhealthy manner and need to be motivated and guided to live healthier. They are lazy and don’t just want to workout for the benefits that a workout will give the body. The program is needed to motivate people to do workouts to live healthier. It is also needed to help guide them to live healthier.

## Possible solution: The program will allocate points based how much effort a person puts into a workout. If they put more effort into the workout, they will earn more points. If they workout more frequently, they will earn bonus points. This will motivate them to workout frequently and put effort into it. They points that they gain; can be exported and then used to buy nice healthy related things. The program will also guide them to live healthier by offering statistics about their workouts. The program will interact with the database by storing the user information and using it for calculations. It will store their workout info and manage their points that they earn from working out and then extracting it for use. The program will interact with text files, using them to store login information and using them to store which workout are available to the user when doing a workout.

## Scope: The program will not be able to force users to workout and live healthier, it can only motivate them to live healthier. The program will not allow for a vast range of activities to be completed in a workout, as only one of each activity type can be done except for strength. Program can’t charge the user actual fees in real life money; It can only charge fake currency and not actually access the bank.

# **Task 1B : User Stories**

| **WHO** | **WHAT** | **WHY** |
| --- | --- | --- |
| Administrator | Add admins | To make it easier to manage the users by having more admins as the program grows in user amounts. |
| Administrator | Delete user accounts | If a user wants their account to be deleted and can’t access it do it themselves then the admin can delete if for them. |
| Administrator | Revoke membership | If a user misbehaves or another reason; then the admin can revoke their membership and they will have to pay to regain access to it. |
| Administrator | Change membership info message | If details about how the membership works changes, the admin can change the info message on the register page where the user can get info about the membership |
| Administrator | Charge users fees for damage or other things that they did. Charge fees to the users account | If the user is at the gym and breaks a piece of equipment, the admin can charge them a fee for the piece of equipment that they broke. A fee can also be charged if the user did something else that is fee worthy like wear inappropriate clothing |
| User | Add a workout | The user wants to earn points; thus, they need to add a workout to earn points on it. When they add they workout, then they will receive points. This will motivate them to do more workouts and live healthier. |
| User | Extract Points | The user wants to extract the points that they earned for their workout to an external platform where they can use it for something like buying healthy food or buying new training shoes that they can use to perform more workouts. |
| User | Pay fees | To be a good person that pays for the damage that they caused. If they for example broke a piece of equipment, then they can pay for it |
| User | Edit account info (Update account info) | They want to edit their name/surname/date of birth if they made a mistake inserting it. They can also edit their height and weight as it changes as time progresses and they become stronger. They will also have the ability to edit their login password |
| User | See their workout statistics | They want to track how they workout , or see which activities they did on a certain day. They might also want to see when they earned the most points so they will be able to see a graph that displays when in the last selected time period {Year/weeks/months} they earned the most points. This can help them to track their fitness and determine which activities are better to focus on if they want to love healthier and earn more points. |
| User | Create an account | The user wants to create an account so that they can track their workouts and earn points on their workouts that they can use for other healthy related things such as buying new sport shoes |
| System Manager | Reset the system. Clears the database; delete all the accounts stored in text files; delete the workouts stored in text files and delete all the admin account. | To clear the entire system if the owner would like to restart the entire system for an excellent reason. To restart the program as if it is brand new. Can also be used if you wanted to transfer the program owner ship and ­management to a new team and don’t want the customers info to be transferred alongside it. |
| System manager | Manage admins and take absolute control over the system | Admins can manage admins, so if an admin deletes the other admins to take control of the system, then that admin can be sorted out by being removed by the system manager. To have the upper hand of control over the system by either the owner of the gym (system) or a trusted third party person |
|  |  |  |

# **Task 1B: User Stories - Acceptance Test**

* A customer (User) can view their outstanding fees such as membership fee or broken product charges
* A user (Customer) can add a workout to earn points on it
* A user can extract their points earned to an external platform where they can use it for something else that that platform allows them to use it for.
* An administrator can search for user in a certain range or that have certain attributes (Such as having a membership) and see their accounts displayed as well aa see statistics about how many users out of the total have these attributes or fall in the range
* An administrator can add other administrators to the program to make it easier to manage the program. They can also remove other administrators.
* A user can create an account that they can use to track their workout and earn points
* A user can update their account information to ensure that they get accurate point allocations.

# **Task 2 : Data Dictionary**

# **Data structures**

## Text files:

|  |  |
| --- | --- |
| **Name of text file** | Stength\_Activities.txt |
| **Purpose** | To store the strength activities and the points for the activity to be used in the program. The program will read from the txt file on startup and populate a checklist box on the workout page. It will also store the points that each activity rep is worth into an array that can be used for further calculations |
| **Format** | <Activity Name>#<Points>  Example: Push Ups#2 |
| **Extract of data** | Push Ups#2  Sit Ups#1  Pull Ups#4  Squats#1 |

## Arrays:

|  |  |
| --- | --- |
| **Declaration** | arrStrengthPoints : array[1..20] of integer |
| **Populate arrays** | Populated from text file when the program starts up. The amount of points that the user will receive for each rep that they do of the strength activities that they selected in the check list box. |
| **Processing** | * Add a value to the array when the admin ads a new activity * Delete a value from the array when the admin deletes an activity * Move values in array to other slot when an activity has been deleted (Move the activities forward to match the list in the check list box)   // When an activity has been added/deleted; require a system restart for the changes to take effect |

|  |  |
| --- | --- |
| **Declaration** | arrStrengthReps : array[1..20] of integer ; |
| **Populate arrays** | Using an input box to receive input from the user. The user will input the amount of reps that they did for the selected activity in the check list box into the input box and it will then be read into the array. This will be stored parallel to the name of the activity stored in arrStrengthNames |
| **Processing** | * Multiply strength activities reps with points per rep to calculate the total amount of point for that strength activity * Sort the reps from the highest to the lowest and display when the user confirms their workout info |

|  |  |
| --- | --- |
| **Declaration** | arrStrengthNames: array[1..20] of string |
| **Populate arrays** | Populated with the selected items from the checklist box. Esch selected item will be stored to the array as a string; parallel to the reps for that activity stored in arrStrengthReps |
| **Processing** | * Find the name of the activity to match the activity in the check list box; used to display the reps in the input box when the user wants to check/change the reps that they inputted (Search with a while loop to find one) * Sort the array for output for the check of the data according to the activity name according to the alphabet. |

## User defined methods

|  |  |
| --- | --- |
| **Signature** | **Explanation** |
| Procedure DeleteAccount(pPrimaryKey:integer); | Goes thru the tables in the database and deletes all the records that use pPrimaryKey as their foreign key; deleted the txt files associated with these records and deleted the login txt file named after pPrimarykey. |
| Function PointMultiplier(pHeight:integer; pWeight:real; pDateOfBirth:TDate):real; | A function that determines a point multiplier. This multiplier will be multiplied to the total workout points to compensate for factors such as a persons age, height and their weight. An older person will have a higher multiplier than a younger person. A heavier to Shorter person will also get a higher multiplier. A Lighter to taller person will get n higher multiplier. This is to ensure that a person who is in their prime does not get huge amounts of points for an easy workout and an elderly person gets more points for easier workouts as it will be difficult for them. |

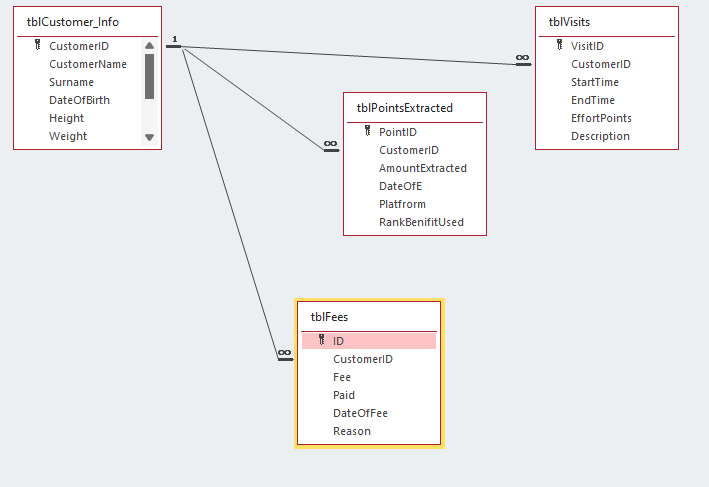
# **Task 3: Database**

## Role of Database: (List of FIVE functions)

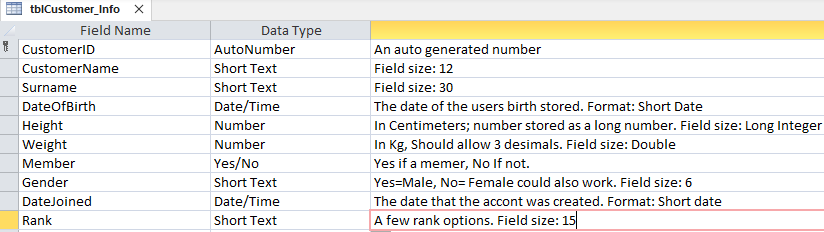
1. The database will Store the user details such as name, surname, age of birth, esc.
   1. This will then be used for calculations such when they do a workout. It will also be used when the administrator wants to find user login details using the users name and/or surname.
2. The database will Store if the user is a member as it will determine if they can earn points on workouts. If a user is a member, then they can earn points which can then be used for healthier living.
3. The database will Track user fees such as unpaid memberships or charges for damage that they did. Track if they have paid for the fee and the fee reason. Allow the user to see their fees and pay them. If the user has to many fees then they will start to earn less points. Their membership can also be revoked if they are not paying their fees. (They will still be listed as a member, however, they will have to pay for their membership again)
4. The database will Track when the customer came to visit the gym. Use this to calculate their time of exercise and store the points that they earned for exercising. Also track allow them to see a description of their specific workout. User can then see stats based on this info and use it to see workout specific info. When a new workout is done, a new record will be added to the visits table.
5. The database will track when the user extracts their workout points. It will allow them to calculate their remaining total points and see when and to where they extracted points.
6. When an administrator is looking for someone, the active selected table with it’s dataset will be displayed in the DBgrid.

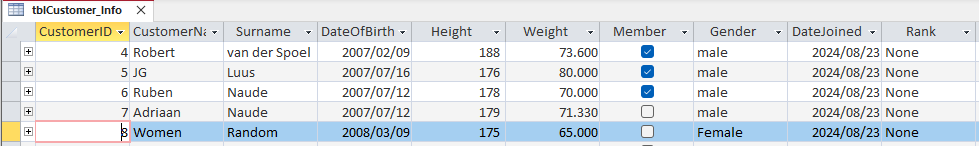
## Database design

Relationship:

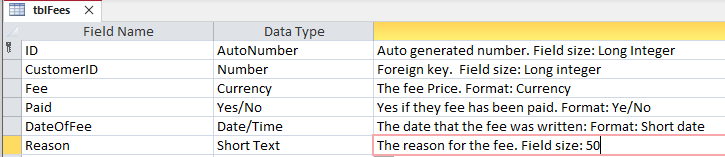


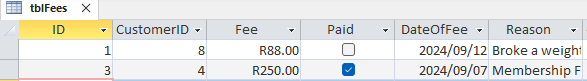
tblCustomer\_Info:



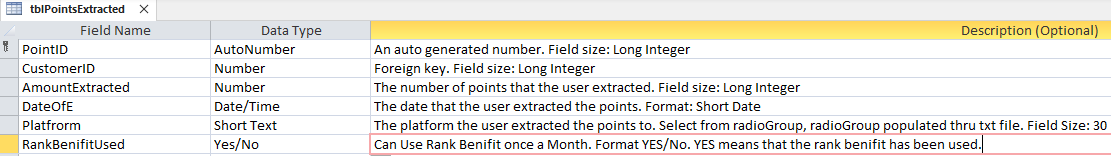


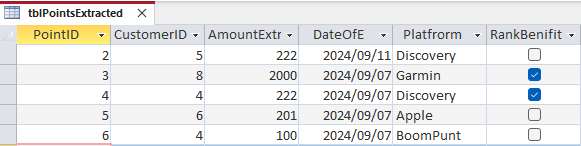
tblFees



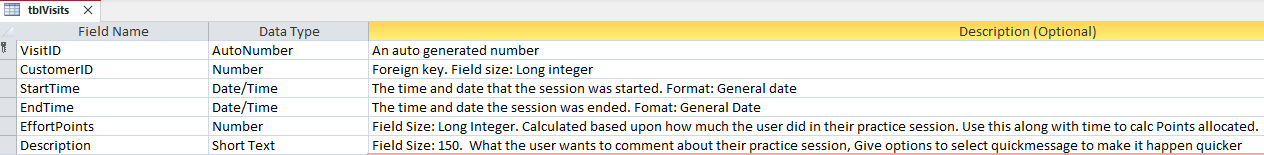


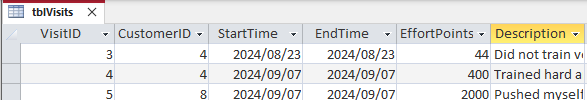
tblPointsExtracted





tblVisits





# **Task 4 : Navigation & GUI Design**

# **Navigation**

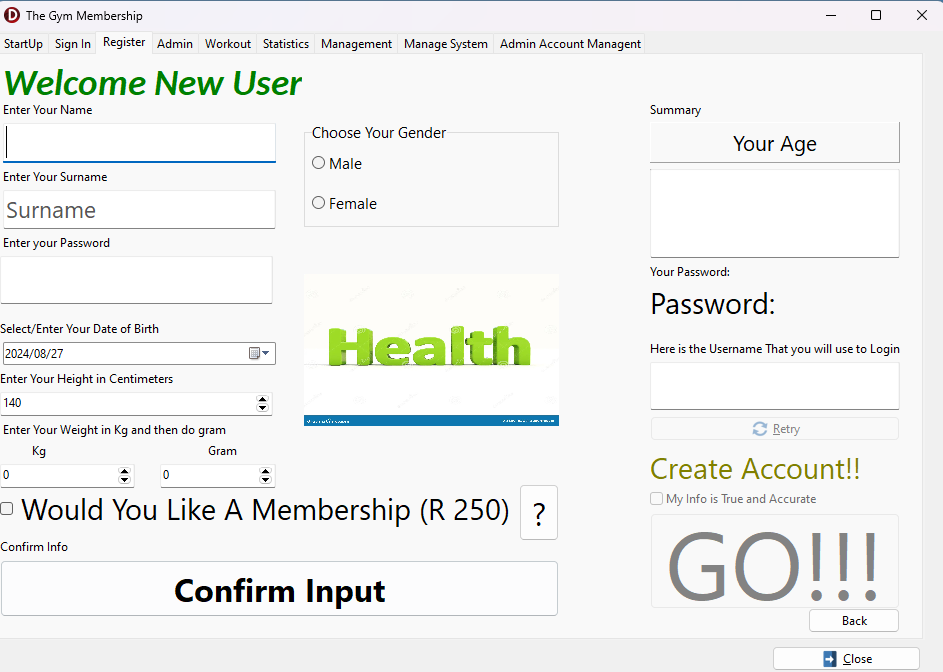
A diagram of a computer flowchart

Description automatically generated

# **GUI Design**

## Tab Sheet 1: Registration

Screenshot of tabsheet 1



## Tab Sheet 2: Admin Page

Screenshot of tabsheet 2

A screenshot of a computer

Description automatically generated

## Tab Sheet 3: Admin Account Management

A screenshot of a computer

Description automatically generated

## Tab Sheet 4: Workout Page

Screenshot of tabsheet 4

A screenshot of a computer

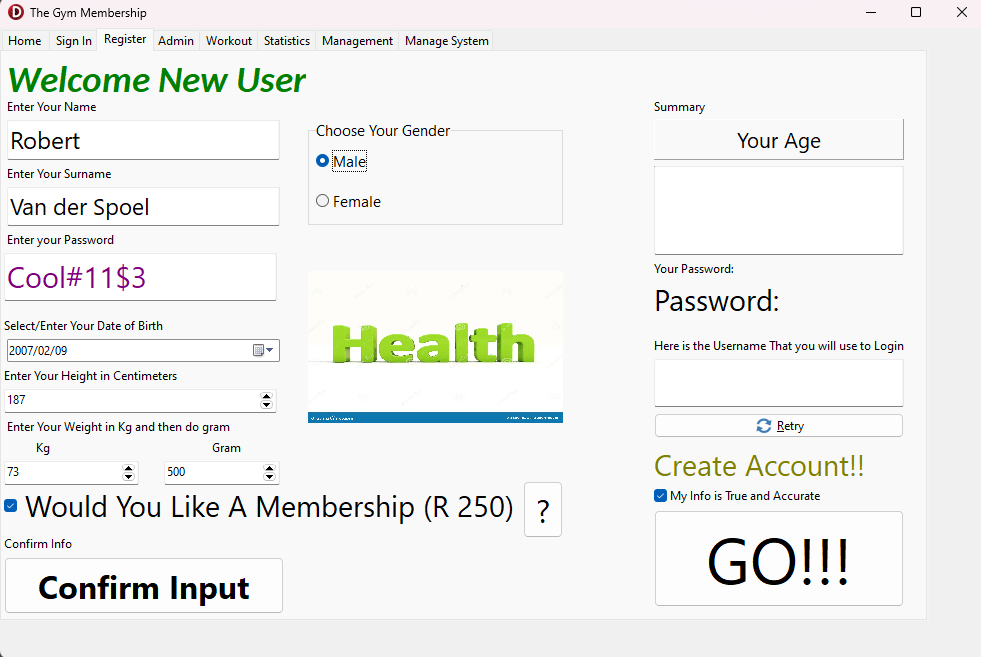
Description automatically generated

A screenshot of a computer

Description automatically generated

# **IPO – Software Design Tool (Task 5)**

## INPUT: TABSHEET 1 **(DO THIS FOR TWO tabsheets) Registration Page**



| **Source** | **Variable name & data type** | **Format (type, size)** | **GUI Object Name** | **Local/Global variable** |
| --- | --- | --- | --- | --- |
| Keyboard | sName; string | Store as a string | edtName | Local |
| Keyboard | sSurname; string | Store as a string that the user entered | edtSurname | Local |
| Keyboard | sPassword; string | A string that the user enters to use as their password.  Format:   * Longer than 8 characters * Shorter that 16 characters * Should contain a number * Should contain a special characters * Should contain a capital Letter | edtPassword | Global |
| Mouse Click | sGender; string | The user selects a Gender from the radiogroup; the gender name is extracted from the radiogroup selected item and stored as a string  When male selected= Person is a male  When Female selected=Person is a female | rgpGender | Local |
| Mouse Click  (Keyboard can also be used to insert the date into the format) | dDateOfBirht; Tdate | Yyyy/mm/dd  Store as a date variable that the user selects from the date time picker | dtpDateOfBirth | Local |
| Keyboard/Mouse | iHeight; integer | Store the height of the person in cm as a integer | sedHeight | Local |
| Keyboard/Mouse | rWeight; real  First Input (Kg) | Store the weight of the person as a real. Use kg. The first input received will be the kg and the second will be the gram | sedKilogram | Local |
| Keyboard/Mouse | rWeight; real  Second Input (Gram) | Store the weight of the person as a real. Use kg. The first input received will be the kg and the second will be the gram | sedGram |  |
| Mouse click | bMembership; Boolean | Select/ unselect checkbox to confirm if the user want s membership. Selected= Yes, want a membership. Unselected= No, does not want a membership  This will be stored as a Boolean | chkMembership | Global |
| Mouse Click | bPay; Boolean | When the GO button is clicked, a dialogue box asking the user YES/NO when they want to pay IF they selected membership  YES for Pay now  NO for pay later  This will be stored as a Boolean | Message Dialogue box | Local |
| Mouse Click | bTrueAndAccurate; Boolean | When the data true and accurate checkbox is clicked then the GO button will be enabled as they confirm that their information is true and accurate. When it is unchecked, the GO button will be disabled as the user is saying that their info is not true and accurate | chkMyInfo | Local |
|  |  |  |  |  |

## INPUT: TABSHEET 2 **(DO THIS FOR TWO tabsheets) Workout Page**

A screenshot of a computer

Description automatically generated

| **Source** | **Variable name & data type** | **Format (type, size)** | **GUI Object Name** | **Local/Global variable** |
| --- | --- | --- | --- | --- |
| Mouse Click | sCardio; string | Get the string from the selected item in the cardio combobox and store it as a string. | rgpCardio | Local |
| Keyboard/Mouse Click | iCardioDistance; integer | Store the distance in kilometres from the cardio activity as an integer | sedCardioDistance | Local |
| Mouse Click | sStrengthAct; string | Store the items selected in the Checklist box as string and read them into an array of string type. Get the name of the selected item from the checklist box. The selected items will then be stored to an array of string type | chkListStrength | Global |
| Keyboard | iStrengthReps; integer | The reps that the user did at each of the Strength Activities will be received as integers using input boxes for each one they selected. The enter the reps as a string that gets converted to an integer when they enter it. It then gets read into an array of integer type | Input Dialogue box | Global |
| Mouse Click | sLowerBody; string | The activity they did that focused on the lower body, selected from the combobox and then stored as a string | cmbLower\_Weights | Local |
| Mouse Click/ Keyboard | iLowerBodyReps; integer | The user will enter the reps that they did for their lower body exercise in the spin edit by typing the integer from their keyboard or using the arrows to move to their reps done | sedLowerBody | Local |
| Mouse Click | sUpperBody; string | The activity they did that focused on the upper body, selected from the listbox and then stored as a string | lstUpperWeights | Local |
| Mouse Click/ Keyboard | iUpperbodyReps; integer | The user will enter the reps that they did for their upper body exercise in the spin edit by typing the integer from their keyboard or using the arrows to move to their reps done | sedUpperBody | Local |
| TextFile | sLowerBodyItems; string | Occupy the combobox by looping thru a txt file and imputing the values into the combobox. Read into combobox as a string | cmbLower\_Weights | Local |
| TextFile | sUpperBodyItems; string | Occupy the lsitbox by looping thru a txt file and imputing the values into the lsitbox. Read into listbox as a string | lstUpperWeights | Local |
| TextFile | sStrenghtItems; string | Loop thru a txt file and add the lines of the txt file to the CheckListBox. Read it into the CheckListBox as strings | lstUpperWeights | Local |

## VALIDATION:

| **Input** | **Validation technique and description** | **Associated error message** |
| --- | --- | --- |
| **Registration** |  |  |
| Name | Name Should not be empty; name not equal to an empty string | Enter a name |
| Name | Length, should be larger than 0 and smaller or equal to 12 characters | Name Length should be between 1 and 12 characters |
| Name | Use pos to ensure that no spaces are present | No spaces are allowed in the name field |
| Name | Check that the entered name only contains alphabet characters using IN[] | Only alphabet characters are allowed in the name |
| Surname | Length, should be larger than 0 and smaller or equal to 30 characters | Surname Length should be between 1 and 30 characters |
| Surname | Check that the first character is a alphabet character and not a space | Begin surname with alphabet character |
| Surname | Check that the entered Surname only contains alphabet characters using IN[]. May also contain spaces | Only alphabet characters and spaces are allowed in the surname |
| Password | * Longer than 8 characters * Shorter that 16 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) | Enter a valid password with a length of 8-16 characters; containing a number, a special character and a capital letter. |
| Date of birth | Check that date is valid; person older than 13 years and younger than 100 years. This will validate using the date variable | Please enter a valid date of birth. You need to be between 13 and 100 years old |
| Gender | Item index not = to -1, ensures that user selects a gender | Please select a gender |
| Yes my Info is True and accurate | Check that the checkbox for info validity has been selected, if not, show error message saying that user should confirm their details. | You need to click the checkbox above to tell us that your information provided is true and accurate |

|  |  |  |
| --- | --- | --- |
| **Workout** |  |  |
| Activity | If statement with ‘ors’. Check that at least one activity has been selected. Could be cardio/strength/weights | Please select an activity |
| Distance for cardio | Check that a kilometre greater than 0 has been inserted into the spin-edit | Please enter your distance for cycling/running |
| Reps for strength training | Ensure that each of the strength activities that was selected has received an input for the amount of reps for that activity. Check that a valid number has been entered in the input box before continuing to the next activity | Please enter a valid number of reps for your () activity |
| Lower Body focused lifting reps | Check that a number greater than 0 has been inserted into the spin-edit for the reps of the lower body focused weight activity | Please enter a valid amount of reps for your lower body focused weight lifting |
| Upper Body focused lifting reps | Check that a number greater than 0 has been inserted into the spin-edit for the reps of the upper body focused weight activity | Please enter a valid amount of reps for your upper body focused weight lifting |

## PROCESSING:

|  |  |
| --- | --- |
| **What processing will be done** | **Algorithms, formulas, etc.** |
| **List:** | **Algorithms:** |
| 1. Adding an activity to the list of activities |  |
| 1. Signing in to the program |  |
| 1. Calculate personal health points   (Based on factors such as weight, age, gender, height and rank) |  |
| 1. Delete an Account |  |
| 1. Calculate workout points 2. (Based on the activity you did) |  |
| 1. Account Recovery |  |
| 1. Create a Username for the user | Algorithm 1:  Name ← input from user  Surname ← Input from user  Year\_born ← Input from user as date of birth  Random\_Number ←Randomly generated 3 digit number  arrayWords ← [‘DOG’, ‘HERO’; LEGEND; TREE]  Word ← arrayWords[random number from 1-4]  Surname ← Uppercase(Surname)  Year\_Born ← YearOf(Year Born )  YearBorn\_String ← YearBorn[3] + YearBorn[4]  Surname length ← Length(Surname)  Username ← Name + Surname[1] + Surname[Surname Length] + YearBorn\_String + Word + random\_Number |
| 1. Search for users in age range | Algorithm 2:  Starting\_Age ← Input from user  Ending\_Age ← Input from user  DateToday ← Todays date  tblCustomer\_Info.First  while not end of tblCustomer\_Info  DateOfBirth ← tblCustomer\_Info['DateOfBirth']  Age ← YearsBetween(DateToday , DateOfBirth)  if (Age >= Starting\_Age) and (age <=Ending\_Age)  Display record info  tblCustomer\_Info.next  End Loop |
| 1. Encryption Algorithm | Algorithm 3:  INPUT← input from the user  ENCRYPT ← ''  Loop thru characters in INPUT  open file 'keys.txt'  while not end of 'keys.txt'  Line ← current line of file  if Line[1] = characters  ENCRYPT ← ENCRYPT + Copy(Line, from place 3, to space 13)  Go to next line in file  End Loop  close 'keys.txt' |
| 1. Check user points | Algorithm 4:  Foreign\_Key ← primary key of active record in primary table  Points\_Recieved ← 0  Points\_Extracted ← 0  tblVisits.First;  while not end of tblVisits  if Foreign\_Key = tblVisits['CustomerID']  Points\_Recieved ← Points\_Recieved + tblVisits['EffortPoints']  tblVisits.next  End Loop  tblPointsExtracted.First;  while not end of tblPointsExtracted  if Foreign\_Key = tblPointsExtracted['CustomerID']  Points\_Extracted ← Points\_Extracted + tblPointsExtracted['AmountExtracted']  tblPointsExtracted.next  End Loop  Total ← Points\_Recieved -Points\_Extracted |

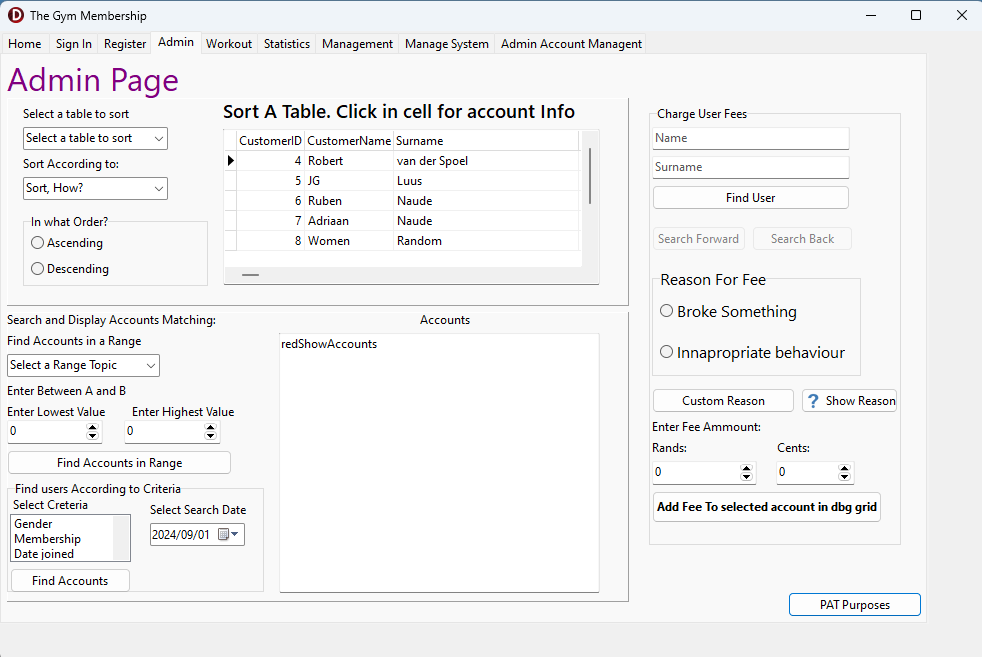
## OUTPUT: TABSHEET 1 **(DO THIS FOR TWO tabsheets) Register Page**

A screenshot of a computer

Description automatically generated

| **Data for Output** | **Format (Type, size)** | **Output Component** |
| --- | --- | --- |
| Age | Calculte the persons age today using their date of birth  [Your age: … years]. Display as integer | pnlAgeCheck |
| Name | The persons name as a string  Name: \_\_\_\_ | redConfirmInfo |
| Surname | The persons surname displayed as a string  Surname: \_\_\_\_ | redConfirmInfo |
| Height | The persons heigh in cm, displayed as an integer  Height: \_\_\_ cm | redConfirmInfo |
| Weight | The persons weight in kg  Kg.gram  Displayed as a real  Example: 55.534 (55 kg; 534 gram)  Weight: \_\_\_ kg | redConfirmInfo |
| Membership | Yes/No Selected  Membership: YES / NO  Display as a string | redConfirmInfo |
| Password | Display the Password; use red colour to attract attention. Display as a string | lblPasswordDisplay |
| Username | Display the generated username of the user in the Colour green. Display as a string data type. Format: Name + First character of surname as a capital + last character of surname as a capital + last two digits of year born + Random word from list + 3 digit generated number | edtUsername |

## OUTPUT: TABSHEET 2 **(DO THIS FOR TWO tabsheets) Admin Page**



| **Data for Output** | **Format (Type, size)** | **Output Component** |
| --- | --- | --- |
| Name, Surname, Date of Birth | Display in the dbGrid, each record under the other, according to the input received. The Customer name, then the Surname will be displayed. The admin can scroll to the right to see more user details | dbgTable |
| Range such as age range, height range, weight range. The data for the output is the specification of that user | The user ID, Name, Surname and then the range, say their age if age was selected or height was selected.  Example:  ID Name Surname Age  22 Robert PietMan 17  ID and Age will be displayed as integers; Name and Surname will be displayed as strings. | redShowAccounts |
| Name; Surname | Displayed in the dbGrid, matching the details that the admin entered in the edits. The active record (Active account) will be the account used for further output | dbgTable |
| Custom reason stored in a variable | Stored as plain text. Maximum size is 50 characters. Received as input, displayed using a showmessage dialogue box, to show the admin the custom message that he/she wants to give as a description to the fee | Showmessage dialogue box |
| Reason for fee; fee amount; name; surname; primary key | Showmessage dialogue box confirming that the fee has feen added to the account. It will be displayed with a showmessage in this format:  ‘Fee added to account {Primary Key } with the holder name of {Name} {Surname }.  Reason: {Reason for fee}  Amount: {Fee amount}’  The primary key is displayed as an int. The fee will be displayed as a number with 2 decimal places. | Showmessage dialogue box |

A screenshot of a computer

Description automatically generated

| **Data for Output** | **Format (Type, size)** | **Output Component** |
| --- | --- | --- |
| Name; Surname; Age; Height; Weight; Date Joined; Found Account ID | Display in the memo as a string using this format:  {Found account ID} {Name} {Surname} {Age} {Height} {Weight} {Date joined}  Will be displayed as a long string. The age and height will be displayed as integers. The weight will be displayed a number with 3 decimal places. The date joined will be displayed as the date; in the format that the active computer displays the date  Example: yyyy/mm/dd | memAccountSearch |
| Found account ID; Name; Surname | A number, starting at 1, that will indicate the account that has been added to the memo. There can be many people with the same names/surname; thus, many will be displayed to the memo. Each one displayed will receive an ID and their name, surname and ID will then be displayed in the list box in this format as a string:  {ID} {Name} {Surname}  The ID will be displayed as an integer and the name + surname will be displayed as strings | lstAccountSelect |
| Username | The username of the selected account; displayed as a string in the label | lblAccountDisplayUsername |
| Password | The Password of the selected account; displayed as a string in the label | lblAccountDisplayPassword |
| Admin Account name | The admin account name displayed in a message box as a string in this format:  {Admin Account Name} has been deleted | Showmessage dialogue box |

Image of a gym, just in case it is not known

