Vanier College

Deliverable 6

  Client: Opeq, Simon

System Development Section 01

Team Orange:

Jiamin Yuan

Dinal Patel

Craig Justin Balibalos

Alihan Djamankulov

Ibrahim Awad

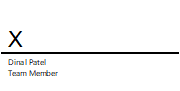
Submission Date:

November 7, 2022

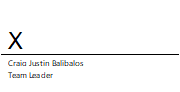
I, (Jiamin Yuan), student ID# (2055624), certify that I have contributed to this deliverable, (signature – this can be a scanned image, or an electronic signature).



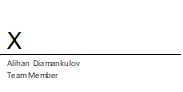
I, (Dinal Patel), student ID# (2042827), certify that I have contributed to this deliverable, (signature – this can be a scanned image, or an electronic signature).



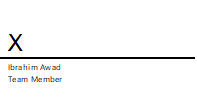
I, (Craig Justin Balibalos), student ID# (2069192), certify that I have contributed to this deliverable, (signature – this can be a scanned image, or an electronic signature).



I, (Alihan Djamankulov), student ID# (2033628), certify that I have contributed to this deliverable, (signature – this can be a scanned image, or an electronic signature).



I, (Ibrahim Awad), student ID# (2032818), certify that I have contributed to this deliverable, (signature – this can be a scanned image, or an electronic signature).



**Table of contents**

Statement ---------------------------------------------- 4

Executive Overview ---------------------------------------------- 4

Business Problem ---------------------------------------------- 4

Narrative Description of the Database Design ----------------------- 5

Appendix 1 ---------------------------------------------- 6-10

Appendix 2 ---------------------------------------------- 10-11

Appendix 3 ---------------------------------------------- 12

Appendix 4 ---------------------------------------------- 12

Bibliography ---------------------------------------------- 13

**Previous Work Statement**

Our team will focus on creating the application from scratch using C# language. Some requirements for the application are that it must be downloadable on PC, and it must be able to connect to a web database. We will use the ideas that we learned from Application Development 1 in the previous semester. We will not use any previous code, but we will use the knowledge that we learned from before.

**Executive Overview**

The problem that the company OPEQ has is storing caller information on an excel sheet. They want us to make an application that will make it more efficient to store caller information.

In the previous deliverable, we made prototypes to show how our application will roughly look and the flow of it. We showed it to our client to get feedback if it is good and if our client wants something changed.

In this deliverable, we did our database design by making a data dictionary to then make an ERD, containing all the basic information like the IDs of the people involved whether they are part of the company or clients, names, contact information and descriptions of the item’s problems as well as the measures applied to fix the problems etc. We looked at the class diagram, that was done in a previous deliverable, to see if we needed to update it. After doing the database, we have the database queries which will help us see if there is anything to optimize as optimizing database queries helps with the efficiency of the application.

**Business Problem**

The problem that our client told us is that they are having a tough time recording the information of the customers that call them. They use an excel sheet to record the information but it takes a while to do it so some information might get mixed up with the others or get forgotten completely. The solution that our client proposed is to make a desktop application that will make recording customer information fast and easy and it will also make the viewing of the records more organized. We made a data dictionary to aid on helping make the ERD. This will help us a lot with the coding and how the database will look like, since the whole application is based on saving records in a database.

**Narrative Description of the Database Design**

The user can login as an employee by entering their name on the login page. Their name will be stored into the database with a unique employee id. They will be able to choose from add data, modify data, or view data. If they choose to add data, they need to input the client's name, client description, date, client email. They can add the client phone number and client address if it is available (nullable). They can select how they contacted the client using radio buttons. If it is by email, telephone or in person. A MAT (item barcode) is required when adding a product. Moreover, they can select the problem from dropdown. Whether it is computer problem, laptop problem, screen problem or phone tablet problem, it will provide different category under type of problems. Once the user login as an admin using name and password, they have the power to add or delete a record of the problem tables. The dropdown field for problem type on the add data page will change following the modifications made by the admin.

When the employee chooses to modify data, they need to enter a valid MAT to access the matching record. It will show the data they have entered on add data page. They need provide some other information in addition to that. Describe in words that the order type and the action they took to solve to problem. The specific date that the problem got solved. Choose between solved, unresolved, and ongoing radio buttons to indicate the status of the present problem (nullable). If the client's phone number is currently available, they can add it (nullable). If it is a new order, it needs the send date and return voucher.

The employee can filter the data by problem status when they decide to view it (solved, unsolved, ongoing). It will show the MAT, client name, and order number for the records that match. By providing the client's name, MAT, or order number, it can locate a particular record.

**Appendix 1**

Data dictionary

Employee table

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Data Format** | **Field Size** | **Description** | **Example** |
| **emp\_id** | int | nnnnnn | 6 | Primary key to differentiate each employee | 000001 |
| **name** | Varchar |  | 50 | Employee name | ‘Simon’ |

Admin table

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Data Format** | **Field Size** | **Description** | **Example** |
| **admin\_id** | int | nnnnnn | 6 | Primary key to differentiate each admin | 000001 |
| **name** | Varchar |  | 50 | Admin name | ‘Simon Provencher’ |
| **password** | Varchar |  | 10 | To store a password | 54321 |

Client table

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Data Format** | **Field Size** | **Description** | **Example** |
| **client\_id** | int | nnnnnn | 6 | Primary key to differentiate each employee | 000001 |
| **name** | Varchar |  | 50 | To input Client name | Alex |
| **description** | Text |  | 1000 | To enter Description of the problem | HDMI cable missing |
| **date** | date | DD/MM/YYYY |  | Date contacted | 18/10/2022 |
| **Phone\_num** | Varchar |  | 20 (Nullable) | To enter the phone number so that the employee can contact them later | (514)123-4567 |
| **email** | text | [name@domain.com](mailto:name@domain.com) | 80 (Nullable) | To enter the email so that the employee can contact them later | Alex@email.com |
| **By\_email** | boolean | 0 | 1 | 1 means the client reached out by email, 0 means they did not | 0 |
| **By\_telephone** | boolean | 0 | 1 | 1 means they came in person, 0 means they did not come | 0 |
| **In\_person** | boolean | 0 | 1 | 1 means they came in person, 0 means they did not come | 1 |
| **address** | text |  | 100 (Nullable) | To enter address in the case of a delivery | 123 Someplace, Somewhere, F9R 2T6 |
| **Computer\_prob\_id** | int | 000003 | 6 (Nullable) | Foreign key for the computer\_prob table to load the drop\_down | 000001 |
| **laptop\_prob\_id** | int | 000123 | 6 (Nullable) | Foreign key for the laptop\_prob table to load the drop\_down | 000001 |
| **Screen\_prob\_id** | int | 000052 | 6 (Nullable) | Foreign key for the screen\_prob table to load the drop\_down | 000001 |
| **Phone\_tablet\_prob\_id** | int | 000036 | 6 (Nullable) | Foreign key for the phone\_tablet\_prob table to load the drop\_down | 000001 |
| **MAT** | varchar | MAT-nnnn-nnnn | 13 | Foreign key for the modify\_data table to load the data | MAT-1022-4568 |

Computer\_prob table

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Data Format** | **Field Size** | **Description** | **Example** |
| **Computer\_prob\_id** | int | nnnnnn | 6 | Primary key to differentiate each problem | 000001 |
| **comp\_desc** | text |  | 100 | A summary of the problem |  |

Laptop\_prob table

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Data Format** | **Field Size** | **Description** | **Example** |
| **laptop\_prob\_id** | int | nnnnnn | 6 | Primary key to differentiate each problem | 000001 |
| **Laptop\_desc** | text |  | 100 | A summary of the problem |  |

Screen\_prob table

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Data Format** | **Field Size** | **Description** | **Example** |
| **screen\_prob\_id** | int | nnnnnn | 6 | Primary key to differentiate each problem | 000001 |
| **Screen\_desc** | text |  | 100 | A summary of the problem |  |

Phone\_Tablet\_prob table

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Data Format** | **Field Size** | **Description** | **Example** |
| **phone\_tablet\_prob\_id** | int | nnnnnn | 6 | Primary key to differentiate each problem | 000001 |
| **Phone\_tab\_desc** | text |  | 100 | A summary of the problem |  |

Order\_Type table

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Data Format** | **Field Size** | **Description** | **Example** |
| **Order\_type\_id** | int | nnnnnn | 6 | Primary key to differentiate each order type | 000001 |
| **name** | text |  | 100 | To specify the type of order that is associated with the product |  |

Action\_Took table

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Data Format** | **Field Size** | **Description** | **Example** |
| **Action\_took\_id** | int | nnnnnn | 6 | Primary key to differentiate each action | 000001 |
| **action** | text |  | 100 | To specify the action took to fix the client’s problem |  |

Product table

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Data Format** | **Field Size** | **Description** | **Example** |
| **MAT** | varchar | MAT-nnnn-nnnn | 13 | Primary key to differentiate each product | MAT-1022-4568 |
| **Product\_name** | text |  | 100 | To specify product |  |

Modify table

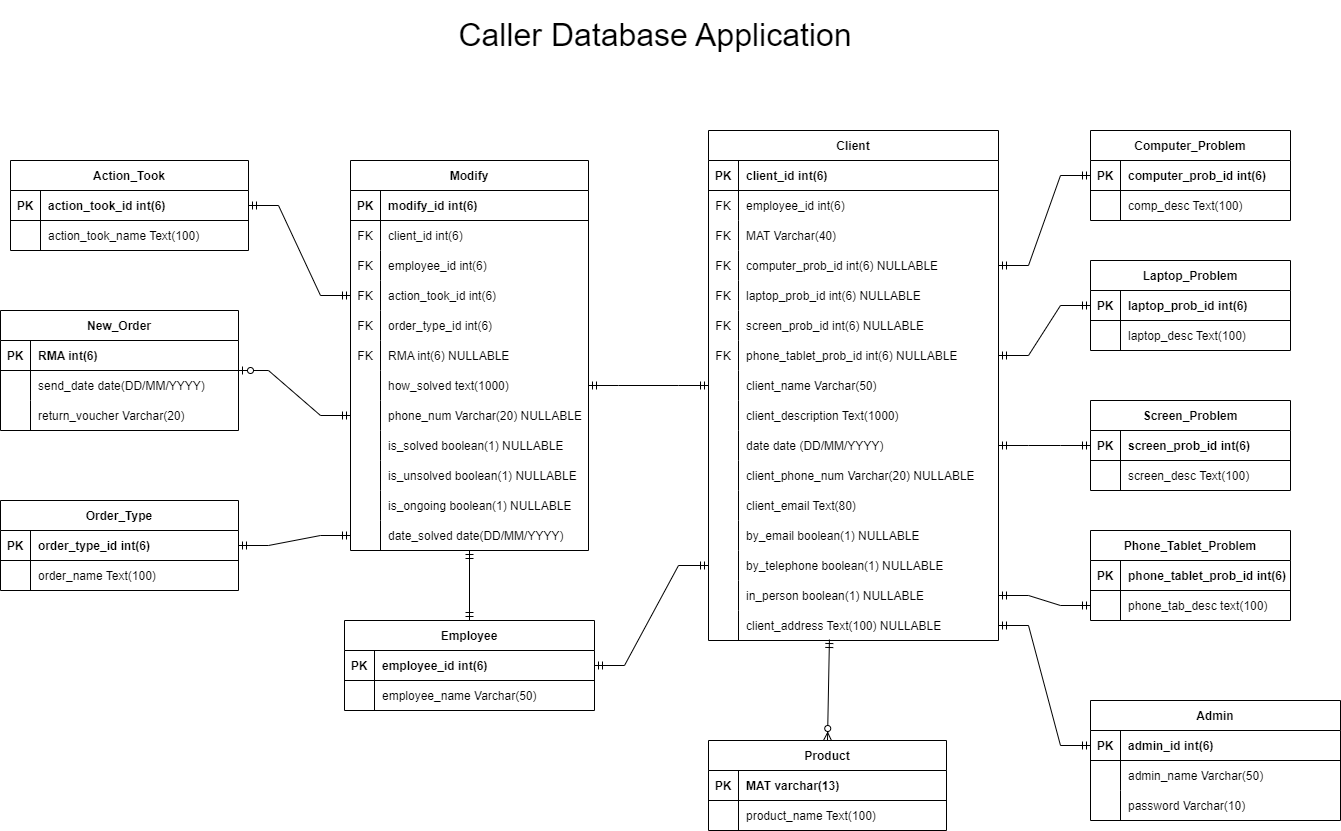
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Data Format** | **Field Size** | **Description** | **Example** |
| **modify\_id** | int | nnnnnn | 6 | Primary key to differentiate each employee | 000001 |
| **Order\_type\_id** | int |  | 6 | Foreign key (primary key of order\_type table) | MAT-1022-4568 |
| **Action\_took\_id** | int |  | 6 | Foreign key (Primary key of action\_took table) | 000001 |
| **How\_solved** | text |  | 1000 | Describe how it was solved | It was solved by ... |
| **Phone\_num** | Varchar |  | 20 (Nullable) | Phone number of the client | (821)211-1232 |
| **Is\_solved** | boolean | 1 | 1 (Nullable) | 1 if the problem is solved. 0 if the problem is in other status | 0 |
| **Is\_unsolved** | boolean | 0 | 1  (Nullable) | 1 if the problem is unsolved. 0 if the problem is in other status | 0 |
| **Is\_ongoing** | boolean | 0 | 1  (Nullable) | 1 if the problem is ongoing. 0 if the problem is in other status | 1 |
| **Date\_solved** | date | DD-MM-YYYY | (Nullable) | Problem solved date | 22/11/2022 |
| **RMA** | int |  | 6 (Nullable) | Foreign key (Primary key of the new order table) | 128723 |

New\_Order table

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Data Format** | **Field Size** | **Description** | **Example** |
| **RMA** | int | nnnnnn | 6 | Primary key to differentiate each new order | 000001 |
| **Send\_date** | Date | DD-MM-YYYY |  | Date the order was sent | 24/11/2022 |
| **Return\_voucher** | Varchar |  | 20 |  | H812VO |

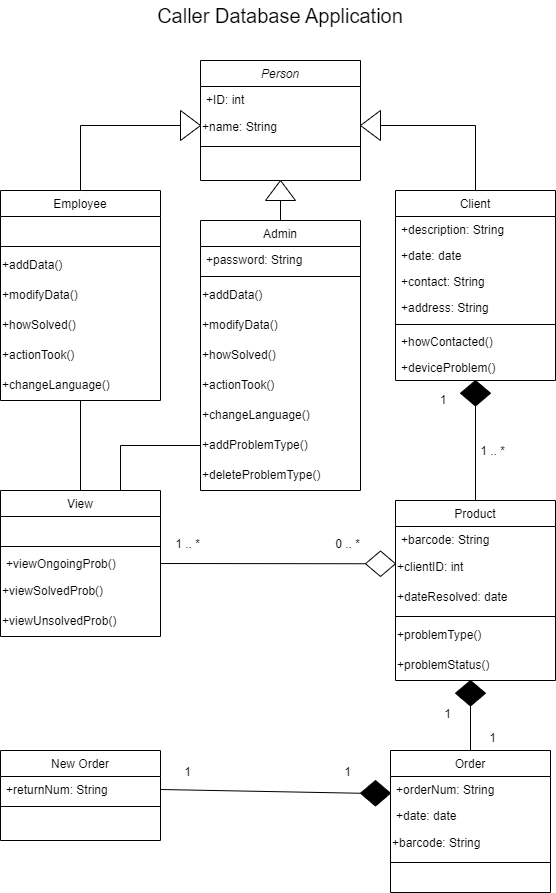
**Appendix 2**

ER diagram



Class diagram

The class diagram is similar to the one from deliverable 3 because we kept the same classes, and the relations are still the same, but it is also different because we added a new class called Admin. The admin class is similar to the Employee class but it has 2 new methods which are addProblemType() and deleteProblemType().



**Appendix 3**

We would need to optimize since all the queries will be used to search up information quite often. All the queries use "SELECT” statements, so we can use indexes to speed up the search. Also, “IN” predicate can be used instead of =, And/or operators. The classes are normalized.

**Appendix 4**

Based on our database size, the access speed will be fast. The database will be frequently accessed because the entire application uses the database. Employee information is stored from the very beginning of the login screen. After logging in, employees can choose to add, modify, or browse data. A response time of three to six seconds will be necessary.

**Bibliography**

Gaudenz, A., & Benson, D. (2000). *Diagrams.net - free flowchart maker and diagrams online*. Flowchart Maker & Online Diagram Software. Retrieved November 7, 2022, from <https://app.diagrams.net/>