Costa Rica Institute of Technology

San Carlos Campus

School of Computer Engineering

Databases, group. 50

First programmed project

“Database of Straightening and painting workshop.”

Students:

Raschell Jarquín Quesada

Fabricio Porras Morera

Carlos Solís Mora

Professor:

Efren Jiménez Delgado

Delivery date:

September 20th, 2022.

Second semester of 2022.

Table of contents:

Executive Summary

The following content of this project, is mainly based on the creation of a database management system, generated from the PostgreSQL programming language. The central focus of the project consists of the development of a software product, that is generated by a request from a determined client.

In the case of this work, it was decided to choose a straightening and painting workshop for cars as the client, to which the database management system will be created. To start with the development of the application, you must first make a requirements assessment, necessary in all types of databases, and it was from this process that many of the necessary specifications were extracted so that the database works perfectly for the client's situation.

Some special features of this database management system are that the workshop does not offer a product for sale as such, like spare parts or oil, since after making an evaluation, a price is agreed between the customer and the workshop, and the products are all bought from a supplier. This database management system also explores the solution to create a correct relationship between the different entities, in order to generate invoices for the different payments that can be made.

Once the requirements that the database will need have been specified, we move on to the second stage of the project, which would be to propose a graphical solution using both the Entity-Relationship model and the relational model. First, it is necessary to make a correct entity-relationship model, this method is used since it is the best way to facilitate the general understanding of the relationships between the different entities and thus get an idea of their behavior once it is done in code. So once the entity-relationship model is perfectly structured, it is passed to the relational model, this with the aim of bringing the scheme closer to something more similar to what a code in SQL language would be; this is essential before starting to work on the code since it serves as a tool to rely on for the correct creation of tables, primary keys and foreign keys within the database.

After this is done, and as the third and last stage of the project, we proceed to the generation of the necessary code to correctly create the required database, basing ourselves on the 2 previous stages to be able to reduce errors as much as possible. In the case of this project, there will be 15 objects per table created, in order to have a good amount of information on which to work and make the different queries necessary to demonstrate the proper functioning of the database management system.

Finally, and as a form of delivery, the scripts used to make the database will be generated, and they will be joined with the different evidence of the taking of requirements, as well as the entity-relationship and relational models to form the project in its entirety.

General Objective

To generate a functional database management system that covers all the requirements specified by a chosen client.

Specific Objectives

To carry out a correct requirement assessment, directly from the chosen client to use them later in the creation of the database management system.

To create an efficient and concise Entity-Relationship model that correctly represents the general functioning of the database management system created.

To create a relational model that correctly represents the Entity-Relationship model previously made, to serve as a tool in the creation of the code of the database management system.

Generation of a functional code that meets all the conditions imposed both by the client and by the project itself in the generation of the database management system created.

Introduction

This project will deal with the creation of a database management system, phrase that when mentioned, comes to mind those systems that are used in supermarkets to process the payments for products at cashiers. But the truth is that, when talking about any business, it is a fact that, in these days, it must have a database to keep track of the inventory and the transactions that are carried out throughout the day.

The importance of databases today is much greater than what can be seen with the naked eye, since for the common person, such as any client, from their perspective a payment they make for a product it's just that, a simple payment, but behind that, there is much more to the companies that make those transactions, because they need a way to save that transaction as information in some way, since all information has to be collected for the business to prosper.

This is when databases appear, and play a leading role in the market, since they are the ideal tools to be able to store large amounts of information efficiently. This information is extremely important for the economy of companies, and its correct collection and processing is an absolute priority, as Roztocki said:

> The expectation of gathering information and the creation of knowledge will accelerate economic development by solving existing problems which was already developed in the past. Information products through information technology are the main resource center for the empowerment of any community by reducing the poverty level which indeed helps us for the economic development. (Roztocki, 2019, as cited in Mallik y Bera, 2021)

And it is that, in effect, making such important decisions that can have such a profound impact on the economy of a company become much easier and more accurate if they are made with an information base behind them that helps to choose what is possibly better.

That is why with this project, it is expected to be able to create a tool capable of being competent in a context as vital as the correct management of the

information of any business. For this, an exhaustive recognition of which can be good target companies will be carried out so that the tool generated from this project is the best possible.

It is expected that the correct application of the different concepts learned throughout the study of PostgreSQL databases will help the software delivered to the client to be competent, and it is expected to satisfy the expectations generated by the same interaction of both parties.

Problem description

It is necessary to make a database management that meets the minimum of 20 tables or entities in it, for each of these tables to be considered as usable entities within the system, they must have a minimum of 5 functional attributes apart from their primary key and any foreign keys they may have. Apart from this, at least 10 relationships must also be made between the different tables of the database management system.

Taking into account all these specifications, the resulting software must be created based on the specific requirements of any given client, this client must be chosen before starting the project by the same developers of the database manager, this client is the one who will decide exactly how the database will work, the different characteristics that it will have, in addition to the relationships in general, so a requirements assessment must be made with this client and the software developers before anything.

Finally, to complete the project in its entirety, an entity-relationship model, a relational model, and the correct generation of the database must be generated from the requirement assessment, and the scripts for the generation of the code will also be delivered, together with evidence of the entire process.

Development

The development of this project, as has already been anticipated, will consist of 4 main stages in which the correct development of a database management system will be deepened. Stage 1, which will consist of the process that was carried out in the requirement assessment to obtain the main business needs of the chosen client; stage 2, in which the different procedures and circumstances that occurred during the process of creating the Entity-Relationship model will be described; stage 3; in which the processes, different circumstances and setbacks that arose during the process of this case the relational model will also be described; and finally, stage 4, in which the process of generating the code to create the different tables and their relationships within the database management system as such will be explained.

Stage 1: Requirement assessment

As a client for the development of the database management system, a straightening and painting workshop for all types of automobiles was chosen. By having a relative of one of the software developers working in the workshop taking the initial requirements was very simple, since it could be carried out using instant messaging applications, such as WhatsApp in this case.

To start the process, the client was first asked to give a general description of the operation of the workshop. With this, it was possible to understand a little the mechanics within the workshop, as well as to reach an approximate maximum of entities that were involved in the business process.

Once there was a general mapping of the workshop's operation, the developers began to ask more specific questions to begin to get closer to a more accurate idea of what the workshop does during the process of attending customers. Valuable information was obtained from these questions, such as the existence of deductibles that apply to cars depending on certain circumstances, such as whether they are insured or not, as well as the existence of providers who facilitate them the different components that the workshop needs to carry out the repair of a vehicle and of processes before of the repair of the car, such as the fact that the price that

the repair will cost must be evaluated beforehand so that the customer can decide if he wants to finally repair his car or not.

Finally, and as the last stage of the requirements assessment, a face-to-face meeting was held in the workshop itself, the last remaining doubts about its operation were resolved in order to start with the second stage, the creation of the Entity-Relationship.

Stage 2: Creation of the entity-relationship model

The development of this model began with the creation of the entities that were initially considered the central ones, which were, the entities: person and vehicle. Once they were being worked on, the possibility of being able to make an inheritance raised, in which the attributes of a person would be inherited from both the clients and the workers, and that a vehicle entity would inherit its attributes from each of the types of vehicles that could exist, such as motorcycles, cars, trucks, etc.

However, when trying again to inherit the employee entity to each type of employee existing in the workshop, a problem was found since these children of the parent entity did not have different attributes from each other; this quickly made us discard this idea, and a solution was proposed, in which a parameterization entity called “type of employee” was used, in which the different jobs that a single employee could have within the workshop would be specified. Using this same logic applied to the employee entity, it was noted that the vehicle entity could not inherit due to the same problem, so an entity called “type of vehicle” was also created, in which in this case, the only type of vehicle possible would be specified.

Once the main structure of our database was obtained, some of the entities with which they could be related began to be created. For example, the evaluation processes of the vehicles between an employee and the client, the insured and the insurance policies that are related to the vehicles, the salary of the employees and finally the suppliers that would also be related to the employees. While these entities were being developed, a clear lack of attributes was noticed in some of them, so the

process to complete the 20 entities was seriously affected, since at first it was thought that this had been resolved but it was not.

Despite these setbacks, the final entities continued to be developed for the vehicle repair process to be completed. For this, the entities: "Purchase order", "spare parts", "claim", “supplier invoice” and "debit note" were created, and in this way the logic was achieved so that the employees can ask a supplier for different spare parts necessary for the repair of the car, as well as the option that the employee could claim if one of the orders arrived in poor condition and a refund was needed.

Finally, the necessary entities were created to make the payment from the client to the workshop, for which a "payment" entity and an "invoice" entity were created, which would have the details of all the spare parts purchased and, in general, the final price.

Once all the entities were done, we decided to try to correct the fact that they did not meet the minimum number of imposed attributes, and we began to create the missing attributes, a method that is clearly not the most efficient and it would be much better to do it beforehand, even before creating the relationships.

Step 3: relational model

Once the entity-relationship model is finished, we proceed to work on the relationship model, which will provide more data, more entities, and a clearer image of what the database will be in PostgreSQL.

We decided to do it in a flowchart in the Dia software, since the entities and their respective attributes can be seen in a more order than in a word or another application. In this part, we start with the principal entities that we make in the entity-relationship model and their respective attributes.

Once this is done, we begin to apply what we learned in class, in terms of relationships between one entity or another, it is known that when only one entity has the cardinality many, that entity must have the foreign key of the other entity. In

some cases, being a ternary relationship, several entities had more than one foreign key.

In the case of many-to-many relationships, other entities corresponding to the relationship of the two entities had to be created, some examples of which are: “can be”, “orders”, “is made”, contains, and others. These entities have two foreign keys, this two are the primary key of the entities that has the relationship.

This new thing that the entity-relationship model doesn´t have, helps to make the database in PostgreSQL, because when we make the tables in the relational model, we can better see the attributes that each entity needs to function correctly.

Step 4: code creation

To begin the process of creating tables, we started with the tables that do not have a dependency on others in their relationships. Subsequently, the attributes were created by classifying their domain (INT, VARCHAR and others), then, separately, the CONSTRAINT restrictions were created (Foreign key, primary key restrictions). Once this was done, the tables were revised to add the CHECKS to put conditions in the attributes and the UNIQUES so that values are not repeated in the attributes of a table.

Once each of the necessary tables in the system with their respective restrictions or conditionals were available, the different relationships were made depending on their cardinality.

Throughout the code generation, some problems arose regarding the relations of the tables and their primary keys. For example, the parent classes, when inheriting their primary key from their children, inherited it as an attribute without restrictions, so it was necessary to create the restrictions again in the child tables, so that the inheritance would work correctly.

Also, when generating the compilation with the query tool, on many occasions the duration of the process was excessively long, even when only a simple table creation or an insert was being executed. From this problem it was concluded that this was happening due to minimal flaws in the code syntax, which at the time of

performing all the comparisons, checks and others in the program, the error obstructed the process and slowed everything down.

Finally, the SERIAL type id meant a problem on many occasions, since for example, when a value was going to be inserted in a table, but it was not executed, the serial index was added anyway, and if you wanted to do the insert again correctly, it had the wrong id.

In any case, following the method of order that was proposed to carry out the creation of tables and their different relations, as well as the correction of each one of those problems described above that arose, the database management system could be created in its entirety anyway, and the necessary scripts for the final delivery could be completed.

Chosen Queries and Why.

1. The first query made in the program is a union between the table "vehicle" and the table "insurance policy". This query was made with the aim of testing the operation of neutral values, for when an object in a table does not have a specific value. So, in this case when a vehicle does not have a related insurance policy, the attributes that refer to the insurance policy table, use the designated neutral value to fill in the spaces that should be NULL with understandable values.

2. The second query chosen to demonstrate the operation of the system is a join between the "person" table and the "phone number person" table. It was chosen to show the correct representation of a relationship from 1 to n, so when you want to see people's numbers, all the numbers belonging to a specific person are displayed.

3. The third query consists of a join between the "supplier" table and the "purchase order" table. This query was chosen to be able to represent the relationship from n to n, in which an intermediate table will be created to store the ids that are being related. In the case of this example, when you want to see the relationship between "supplier" and "purchase order", a table will be displayed showing all the ids that are related to each other, without there being any restriction in the fact that several suppliers can share the same purchase orders

4. The fourth query made to check the proper functioning of the created system is the union of the "payment" table with the "invoice" table. This query was chosen to represent how a 1-to-1 relationship between tables works. So, when you ask to see the payments related to its specific invoices, these payments can only have one related invoice, since only one invoice must be generated in the payment process.

5. The fifth and last query generated to check the proper functioning of the tables and their relations is the union of the "supplier" table and the "debit note" table to reach the "employee" table. The idea is to represent how is the process to get from one table to another, using a third table as a bridge, so, in the case of this example, you need to get from a supplier to an employee, passing through a specific debit note that stores both ids, and in this way you can obtain all the employees to whom a specific supplier has made a debit note.

Review of Tables, Their Functionality and Normalization Rules. Next, an analysis will be carried out for each of the tables or entities of the database management system created. The normalization rules that each of the system tables comply with will be indicated, and their general functionality in the system will be mentioned.

"person" table. (The table satisfies the first, second, and third normalization rules). There is a doubt that the first normalization rule with the telephone attribute is broken, however, when declaring it as a multivalued attribute and then creating a table, it complies. The main function of the person table in the system is to be a parent class that inherits the general attributes to the "employee" and "customer" tables.

"employee" table. (The table satisfies the first, second, and third normalization rules). The operation of the "employee" table in the system is very extensive, first of all, the workshop offers it a monetary remuneration for its services after being assigned a task with a client's car. Employees can make valuations to cars, they are responsible for making purchase orders to different suppliers, as well as making claims, if a supplier invoice is incorrect, it is also the one that receives

and processes the debit notes. The employees are related to an employee type parameterization table where their job is related to them.

“customer” table. (The table satisfies the first, second, and third normalization rules). The employees are the ones who own the cars that are processed in the workshop, they are in charge of carrying out the valuation together with the employees, and finally, they are the ones who must make the payment to the workshop after the entire process, they also receive invoices after payment.

“salary” table. (The table satisfies the first, second, and third normalization rules). The salary table stores all the salaries that belong to different employees of the workshop.

“vehicle” table. (The table satisfies the first, second, and third normalization rules). The vehicles are the entity, which is processed in the workshop, the cost of its repair is evaluated in valuation, and may or may not be related to an insurance policy that will mean a variation in costs in valuation. They are related to a "type of vehicle" parameterization table that indicates what type of vehicle it is.

“insurance policy” table. (The table satisfies the first, second, and third normalization rules). Insurance policies play a more basic role within the system, usually as triggers of certain situations, vehicles may or may not have an insurance policy, which belongs to a specific insurance company.

“deductible” table. (The table satisfies the first, second, and third normalization rules). Deductions are applied to a valuation in case the valuated vehicle has an insurance policy; the deductible belongs to a specific insurance company.

“insurance company” table. (The table satisfies the first, second, and third normalization rules). Insurance companies are in charge of providing insurance policies to vehicles, as well as the entities to which the percentage for the deductible applied to the valuation price belongs, in case it is made to a vehicle insured by them.

“valuation” table. (The table satisfies the first, second, and third normalization rules). It is a key entity in the development of the processing of a vehicle, the valuation is carried out with an employee and a client, and in case the processed vehicle is insured, the evaluation is related to a form. The valuation has a tax and a deductible if the vehicle is insured, generates a purchase order once made and is also the one that receives the payment.

“form” table. *(The table satisfies the first, second, and third normalization rules).* The forms are applied in case an insured vehicle is being evaluated in valuation, so valuation may or may not be related to a form.

“task” table. *(The table satisfies the first, second, and third normalization rules).* The tasks work in the system to keep a record of the different jobs that are carried out in the day, they are related to the car in which they are working, as well as the employee who will be in charge of it, in the specific job in which he or she will.

“purchase order” table. *(The table satisfies the first, second, and third normalization rules).* Purchase orders are placed by an employee to a supplier, they are related to a specific evaluation performed on a vehicle and have a "purchase order detail" that belongs to them.

“purchase order detail” table. (The table satisfies the first, second, and third normalization rules). The purchase order detail is the one that has the value of all the products purchased, the quantity of products, the total and the subtotal of a specific purchase order, it is related to the spare parts that the employee wants to buy. A tax must be applied to some of the prices it contains, to which the purchase order detail is related.

“spare parts” table. (The table satisfies the first, second, and third normalization rules). The spare parts are the products that the workshop buys to carry out the repair of an evaluated car, the spare parts that the workshop wants to buy are specified with their relationship in the "purchase order detail" table.

“*supplier” table.* (The table satisfies the first, second, and third normalization rules). Suppliers play a fundamental role in the process of repairing a vehicle since

they are the ones who sell the necessary spare parts. They receive a purchase order from a specific employee and return a supplier invoice to confirm the purchase of the parts; in the event that an employee considers that there has been an error in any of the deliveries of the products, the supplier may receive a complaint about any of their supplier invoices, so to complete the appeal process they may return a debit note that will specify the monetary value of the damaged products so that they can be redeemed at the supplier's store within a deadline.

"supplier invoice" table. (The table satisfies the first, second, and third normalization rules). A supplier invoice is delivered after the supplier receives a purchase order from one of the workshop employees, it contains details such as the total and subtotal value, as well as the type of currency that will be used in the transaction and the date of the transaction.

"claim" table. (The table satisfies the first, second, and third normalization rules). A claim is made by a specific employee of the workshop when a supplier invoice had some type of error, this is made directly to the supplier involved.

"debit note" table. (The table satisfies the first, second, and third normalization rules). A debit note is generated by a supplier after it receives a claim from one of the workshop employees about one of its supplier invoices. The debit note specifies the total and subtotal of the price to be returned, as well as the date of creation and the deadline to redeem this debit note in the supplier's business.

"payment" table. (The table satisfies the first, second, and third normalization rules). A payment is made by one of the clients when the repair of their vehicle has been carried out by the workshop, the payment is made to a specific evaluation, and the payment details the type of currency in which it will be made, the date Initial and deadline for payment

"invoice" table. (The table satisfies the first, second, and third normalization rules). The invoices are generated from a payment made to the workshop for an evaluation and work correctly carried out, the invoice is delivered to the specific client who made the payment, and in this way, the process of the business-client

relationship that occurs in this straightening and painting workshop is completed in its entirety.

Minor tables that are not considered valid entities: “type of vehicle”, “type of employee” and “tax”. (They all satisfy the first, second, and third normalization rules). These tables were not specifically analyzed, since they are not considered valid tables for this project, since they do not meet the minimum number of necessary attributes. Its function within the program is to classify other tables in the case of the "type of vehicle" and "type of employee" tables, and to apply changes to the different prices that are determined, throughout the process that is made by the workshop as is the case of the "tax" table.

Conclusions

While the project was being developed, it was concluded that the more complete the requirements assessment is, the more specific and complex the proposed solution for the creation of the database management system will be, since each of the requirements is one more condition for the code to work exactly as the client is looking for.

At the time of proposing a solution to the requirements requested by the client, it was concluded that the use of the entity-relationship model can be a double-edged dagger, since in one hand, it is very useful to obtain an overview of what will happen in the code, but, in the other hand, if you don't have a good order, it can make the overview very confusing and can lead to errors in the subsequent creation of the relational model.

At the time of generating the code for the creation of the database management system, it was concluded that the relational model is a useful tool to follow the correct order of the creation of tables and their relations since having a structure as like how the creation of tables is programmed in PostgreSQL, programming results intuitive and easy to use when using it as a reference.

Finally, and again in the process of generating the code for the creation of the database management system, it was concluded that the generation of entities as well as their relationships is really simple and completing the minimum is not a problem, however this is not the same with the creation of attributes for each of the tables, since there are many entities that are not so easy to associate at least 5 attributes to them, and this can lead to problems in terms of table count, since you may not have as many valid tables as you thought at first.

Recommendations

Even though a very extensive requirements assessment can generate a more complicated solution, it is recommended to try to make it as complete as possible, since in this way, the database management system that will be generated will be much closer to what the client's business needs and problems such as misunderstandings or decisions by the software developers will be avoided.

When making an entity-relationship model before creating the database management system, it is recommended to have a detailed order when positioning both the entities and their attributes, as well as their relations, to avoid later confusion as much as possible, for this it is also recommended to use annotations that help understand why certain things exist within the model.

To achieve a much more orderly and precise generation of the database management system code, it is recommended to previously create a relational model to use it as a reference in the process; to do this, the relational model must not contain any errors that could lead to a bad code generation, so it is also recommended to review the model thoroughly so that there are no problems.

To avoid having an incorrect notion of the valid number of tables in cases like the ones in this project, in which it is necessary to have a minimum number of attributes per entity to consider them as such, it is recommended once a new entity is proposed, immediately create the necessary attributes and not leave that task for later, this will also be useful to get an idea of how far that entity can be used within the database, and thus new ideas may occur to implement during the process.

Bibliography

Mallik, S., & Bera, D. (2021). Importance of Information Product for Economic Development. Recuperado de: https://www.researchgate.net/profile/Supriya-Mallik/publication/352401112_Importance_of_Information_Product_for_Economic_Development/links/6245848b7931cc7ccf07f15d/Importance-of-Information-Product-for-Economic-Development.pdf

Annexes

Annex 1.

Activity number	Date and hour	Place	Participants	Matters to be discussed	Agreements and responsibilities	Pending issues
1	Tuesday, August 30 at 4:10 p.m.	Instituto Tecnológico de Costa Rica.	Fabricio Porras Morera y Carlos Solís Mora.	Requierments assessment	It was agreed that both responsible, Fabricio Porras and Carlos Solís, would take requirements from the straightening and painting workshop where an aunt of Fabricio Porras works.	Several questions remain to be asked about the business requirements for the correct creation of an Entity-Relationship diagram. Completed? Yes(X) No()
2	Saturday, September 3 at 6 p.m.	Discord and Whatsapp.	Fabricio Porras Morera y Carlos Solís Mora.	Creation of the Entity-Relationship model.	It was agreed that both responsible, Fabricio Porras and Carlos Solís, would create the Entity-Relationship model in the software used in the DIA class, based on the requirements collected to date.	It is pending to consult the professor about several dilemmas about the functioning of inheritance in the created Entity-Relationship model. Completed? Yes(X) No()
3	Tuesday, September 6 at 4:30 p.m.	Instituto Tecnológico de Costa Rica.	Fabricio Porras, Carlos Solís Mora, y Raschell Jarquín.	Optimization of the Entity-Relationship model created.	It was agreed that those responsible, Fabricio Porras, Raschell Jarquín and	It is pending to continue the Entity-Relationship model. Completed?

					Carlos Solís, would carry out the changes proposed by the professor.	Yes(X) No()
4	Friday, September 9 at 8:00 p.m.	Discord and Whatsapp.	Fabricio Porras y Carlos Solís.	Continuation of the Entity-Relationship model created.	It was agreed that the people in charge Fabricio Porras, Raschell Jarquín and Carlos Solís would finalize the Entity-Relationship model after making the last queries about the operation of certain things in PostgreSQL.	It remains to consult the teacher about the creation of some tables, and the operation of some relationships in PostgeSQL programming to finalize the Entity-Relationship model. Completed? Yes(X) No()
5	Tuesday, September 13 at 4:30 p.m.	Instituto Tecnológico de Costa Rica.	Fabricio Porras, Carlos Solís Mora, y Raschell Jarquín.	Continuation of the Entity-Relationship model, creation of the relational model and creation of the PostgreSQL database.	It was agreed that those responsible Fabricio Porras, Raschell Jarquín and Carlos Solís, together, would be in charge of finalizing the 2 models necessary for the project, and would create the	The total completion of the project is pending, including its written part. Completed? Yes (X) No()

					database from them.	
6	Friday, September 16 at 8:00 p.m.	Discord and Whatsapp.	Fabricio Porras, Carlos Solís Mora, y Raschell Jarquín.	Completion of the Entity-Relationship model.	The entity-relationship model was finalized, it was agreed that those responsible Fabricio Porras, Raschell Jarquín and Carlos Solís would connect the following day to finalize the relational model and tentatively begin the creation of the PostgreSQL database.	The relational model and the PostgreSQL database are pending, as well as a part of the written report. ¿Completed? Yes(X) No()
7	Saturday, September 17 at 4:30 p.m.	Discord and Whatsapp.	Fabricio Porras, Carlos Solís Mora, y Raschell Jarquín.	Completion of the relational model, beginning of the generation of the database.	The relational model was finished, the generation of the database began, it was agreed that Carlos Solís and Raschell Jarquín would be in charge of generating the 15 respective data for each table, while Fabricio Porras would be in charge of generating the structure of the database.	The completion of the PostgreSQL database, each table value, and the written report are pending. Completed? YesX) No()

8	Sunday, September 18 at 7:00 p.m.	Discord and Whatsapp	Fabricio Porras, Carlos Solís Mora, y Raschell Jarquín.	Completion of all data for each table	Those responsible Raschell Jarquín and Carlos Solís finished the values of each system table, it was agreed that Fabricio Porras would be in charge of adding this data in the created structures of the code, as well as the final code	The completion of the PostgreSQL database and the written report are pending. Completed? YesX) No()
9	Monday, September 19 at 7:00 p.m.	Discord and Whatsapp	Fabricio Porras, Carlos Solís Mora, y Raschell Jarquín.	Project completion	Those responsible Fabricio Porras, Raschell Jarquín and Carlos Solís finished all the remaining parts of the project	There is nothing pending. Completed? YesX) No()

Annexe 2

https://www.youtube.com/watch?v=_4sSFo-wsWg

Annexe 3

Annexe 4