Object-Oriented Database Edit and Analysis System

NUNO GOMES, Instituto Superior Técnico, Portugal
PEDRO SOUSA, Instituto Superior Técnico, Portugal

Nowadays possessing data is essential for companies, scientist, governments and for that it is really important to have efficient databases to store that data. There are different types of databases, all have the same goal to store data, but with different methods and with it's advantages and disadvantages. In this thesis the main goal was to develop a system to edit and analyze an object-oriented database, to implement this system I used Excel with the help of VBA and some Python scripts to interact with the API which therefore interacts with the database. The database is printed in the Excel sheets, every sheet is a different class, every row an object and every column a property of the object, the user can than edit the cells and update the database. It is also possible to create charts with the data from the database so the user can analyze the data. After testing I was able to realize that Excel is a very good program to edit a database due to its already implemented features to edit data simply and it's very intuitive tabular view of the data.

CCS Concepts: • **Object-Oriented Database System** → **Database Edition System**; *Database Analysis System.*

Additional Key Words and Phrases: Databases, Excel System, Classes, Objects, Relations

ACM Reference Format:
Nuno Gomes and Pedro Sousa. 2022. Object-Oriented Database Edit and Analysis System. *ACM Trans. Graph.* 1, 1, Article 1 (October 2022), 7 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

The need to save data—data about us, data about our surroundings, and data about other species—has grown exponentially with the use of technology in our daily lives. To store this data, databases are used. The data is stored in tables with rows for the number of records in the table and columns for the properties we want to save for each record.

For a simple and straightforward interface with a database, having a system where it is possible to import a database and analyze the data, produce charts and tables, and even be able to change the database in a inline tabular format, is extremely valuable.

2 PROBLEM DESCRIPTION AND GOALS

Link Consulting, the company with which I did this thesis in collaboration, has a product called ATLAS, this product is used to help other organizations manage their projects and build an architecture for the organizations. ATLAS has a timeline that helps the organizations seeing how the projects will develop in the future, it has a time bar that can be moved to go in the past and future and see how the projects are predicted to evolve and how they were in the past compared to the present. ATLAS is also very good at collecting information from internal and external sources.[ATLAS 2022]

ATLAS uses a database with an object-oriented interface. There are repositories that serve as the database, each repository has classes, each class has properties that the class's creator has defined, it is possible to create objects that are associated with the class, an user can then fill the properties with values.

To edit the database the user has to use the ATLAS website which permits the user to edit class by class and object by object which was not very practical, for this reason it was realized that it was necessary to create a new system to edit the database. Another issue that needed to be resolved was finding a system where it was possible to create charts, tables, or more statistical data in order to make reports simpler and with more details. This was crucial because ATLAS assists organizations in project management, so producing reports with statistics about their strategies is really helpful.

After some research about tools to solve these problems, I did not find any already developed tool that would solve these problems, so after some pondering, we decided the best tool to start with would be Excel has it has a tabular view to use and edit the database and it has a lot of features to help edit and analyse the data there.

As goals the main objectives of this thesis was to make a system that would be able to import a database, so the data in the database could be analysed, having the possibility to create charts about the data, and also having a system where the user can edit the database in a simple way.

These were the starting objectives of the system:

- **Importing classes from a repository**: Every repository has multiple classes, the idea is to import each class into a different Excel sheet, and every row represents an object and every column a property of the objects.
- **Edit Data and then update the database**: To edit the data, the user can change the cells with the properties values and then have a button that, when pressed, would see what was changed and update the database.
- **Create charts to analyze the data**: Using Excel, it is really easy to create charts of data, using the Excel interface the user chooses the chart type and then selects the cells with the data to do the chart.

This goals were the initial ones, with the advancement of the development and with some feedback sessions the goals were upgraded.

Authors' addresses: Nuno Gomes, nuno.rodrigues.gomes@tecnico.ulisboa.pt, Instituto Superior Técnico, Av. Rovisco Pais 1, 1049-001, Lisbon, Portugal; Pedro Sousa, Instituto Superior Técnico, Av. Rovisco Pais 1, 1049-001, Lisbon, Portugal, pedro.sousa@linkconsulting.com.

0730-0301/2022/10-ART1 $15.00
https://doi.org/XXXXXXX.XXXXXXX

3 TYPES OF DATABASES

There are several types of databases, every type has the same goal to save data as efficiently as possible, but with different methods that have their advantages and disadvantages. Here I will mention the two types that are more important for this thesis, the relational databases and the object-oriented databases.

3.1 Relational databases

The most popular type of database, it stores data in tables, each row is a record with a key (a unique identifier), and each column is utilized to store a value. It employs SQL to communicate with the database. The keys used to identify the records in the tables make the relationships between the data easier to identify. It has the advantages of being very good at categorizing data, being able to change the database without having to restart everything or change any applications, being simple to use due to using a common language to interact with the database, has security because you can restrict who can access the database, and having multiple simultaneous access from different users. The database is inflexible because new data must be added by updating the database's scheme model, and it is not very scalable because it performs poorly when used with multiple servers. These drawbacks include the need for meticulous planning, the requirement for developers to maintain and optimize the database. [Relational Database 2022]

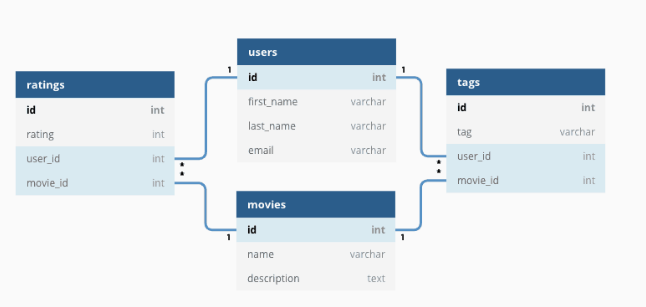


Fig. 1. Relational Database Model

3.2 Object-oriented databases

A database created in object format makes it simpler to call an object and allows for the use of references to refer to other objects. When using object-oriented languages like Java, the objects can be extremely complex and have a ton of properties and methods. Because of this, we can store the entire object in the database and retrieve it whenever we need it, whereas when using relational databases, we would have to break the object down into its variables. As advantages the code becames much cleaner and easier to use when doing a system with an object-oriented database because the developer inserts the object in the database and later can retrieve it from the database and continue using it, as disadvantage it is slower doing a simple lookup in the database, and there is not a universal language like SQL to store the data, every language might have it's own syntax for the database usage. [Object-Oriented Database 2022]

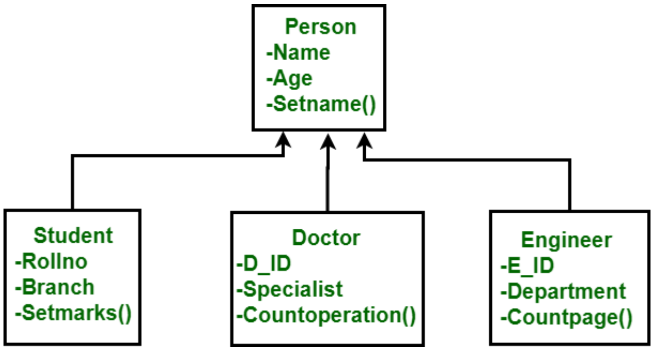


Fig. 2. Object-Oriented Database Model

4 RELATED WORK

As mentioned in the Problem Description, I looked for tools and prototypes that could solve this problem, and I was not able to find any that could do it, the main reason for that is the fact that it was needed to edit an object-oriented database, which as I mentioned before is not as popular as the relational databases, I was able to find tools to solve the problem if the database was a relational database. Here I will show some of these tools that I was able to find:

4.1 Edit a Database Inline

There are some tools that let the user edit a database inline, which means that it will open the database in a tabular view and allow the user to edit directly there.

- DbVisualizer [DbVisualizer 2022]
- SQLGate [SQLGate 2022]
- Postico [Postico 2022]

4.2 Create a chart from the Database

There is also some tools that can create charts from SQL databases, an example of one is Superset, to obtain the data that will subsequently be utilized to produce the charts, the user can connect to a relational database and run SQL queries. A data set that may be used to make charts can be created by the user by selecting columns from tables. This application allows for the creation of a wide range of chart types, including simple and complicated charts, maps, 3D charts, and more. [Superset 2022]

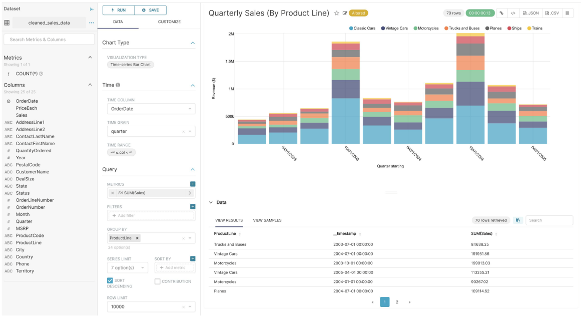


Fig. 3. Example of creating a chart in Superset

5 FINAL GOALS

After all the development was done, the feedback from the company employees was received and the changes implemented, these are the goals of the system and the actions the system is capable of doing:

- Importing all classes and properties from a repository
- Importing some chosen classes and properties from a repository
- Edit Data and then update the database
- Create new objects
- Create new properties
- Create charts to analyze the data

It is possible to see that the initial goals were all fulfilled but they were upgraded with some extra ones, that allowed the user to do more to edit the data in a simpler way.

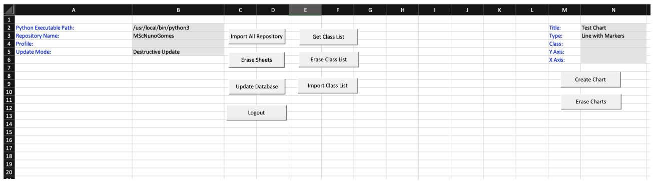
6 EXAMPLE OF USAGE

6.1 Instalation

To install the program, a Python script called install.py is used. It will ask the user for a directory to install the program in, it will build a shortcut in the Desktop for the user's convenience and install the necessary files as well as the Python pip libraries necessary for the Python scripts to run.

6.2 Control Panel

The Control Panel is a sheet that is used to interact to control how the system is used, it has the buttons and the inputs that are going to be used to do all the actions mentioned in the section before, there are some inputs that are obligatory for the system to import a repository, one is the Python executable path, so the Excel can call the Python scripts necessary, the repository name so the system knows which repository to import, than there is the update mode which is used when updating the database with the editions made, there are various modes, ones just to edit properties already with values, other just to edit empty properties and many others modes.

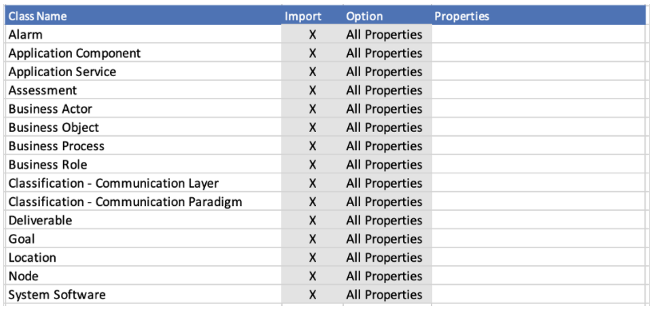


Python Executable Path: /usr/local/bin/python3
Repository Name: MScNunoGomes
Profile:
Update Mode: Destructive Update

Title: Test Chart
Type: Line with Markers
Class:
Y Axis:
X Axis:

Fig. 4. Control Panel Interface

To choose just some classes and properties it was created a table that allows the user to choose which classes and properties he wants to import, it has a column with the name of the classes, there is a column called Option with three options: *All Properties*, *Only These* and *Except These*, with *All Properties* it will import all properties of the class, with *Only These* it will import only the ones on the list to the right, and with *Except These* it will import all but the ones on the list to the right and there is a column called Properties which is will receive the name of the properties in case of the user just wanting some specific properties.

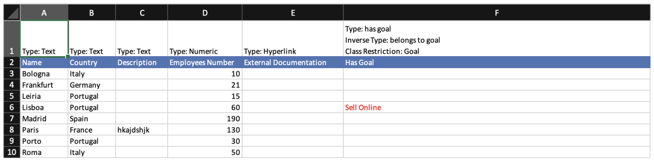


Class Name	Import	Option	Properties
Alarm	X	All Properties	
Application Component	X	All Properties	
Application Service	X	All Properties	
Assessment	X	All Properties	
Business Actor	X	All Properties	
Business Object	X	All Properties	
Business Process	X	All Properties	
Business Role	X	All Properties	
Classification - Communication Layer	X	All Properties	
Classification - Communication Paradigm	X	All Properties	
Deliverable	X	All Properties	
Goal	X	All Properties	
Location	X	All Properties	
Node	X	All Properties	
System Software	X	All Properties	

Fig. 5. Class List Table

6.3 Editing Data

The data will be separated by sheets, every sheet represents a class, the first row of each sheet is the type of the properties, and the second row is the name of the properties, all the rows bellow will have the data of the objects, an object by row, every column has a different property, the text that has a red font means that it is a reference to another object.



	A	B	C	D	E	F
1	Type: Text	Type: Text	Type: Text	Type: Numeric	Type: Hyperlink	Type: has goal Inverse Type: belongs to goal Class Restriction: Goal
2	Name	Country	Description	Employees Number	External Documentation	Has Goal
3	Bologna	Italy		10		
4	Frankfurt	Germany		21		
5	Leiria	Portugal		15		
6	Lisboa	Portugal		60		Sell Online
7	Madrid	Spain		190		
8	Paris	France	hkajdshjk	130		
9	Porto	Portugal		30		
10	Roma	Italy		50		

Fig. 6. Example of an imported class

It is then possible edit the cells adding new values or editing the cells already filled, it is really important to choose the correct update mode, because if you select the incorrect one, you could end up wiping out the majority of the data. You should also double-check because certain update modes just allow you to add new data or merely allow you to amend data that has already been filled in.

6.4 Create Objects and Properties

It is also possible to create objects and properties, if creating a new object it is necessary to add a name value to the row after the last one that has a value, that will create an object with that name, if the user wishes he can already add more properties to the created object. To create the properties it works the same way, the user must insert a type and a name for the property in the cells of the column after the last property inserted.

6.5 Create Charts

The system has the functionality of creating charts, in the Control Panel Figure in the right side we can see some inputs to create a chart, the Title is the title of the chart, the Type is the Type of chart, then chooses the Class he wants the properties from, after that two dropdowns will appear with the properties of the class so the user can choose which values to use.

- Title: Test Chart
- Type: Line With Markers

- Class: Location
- Y Axis: Employees Number
- X Axis: Name

Using these inputs, these would be the result, we can see the title and the axis names are automatically filled with the inputs, the data will be collected directly from the cells, it uses the data that we can see in the Figure that shows the example of an imported class using the column Employees Number for the Y axis and the Name of the Location in the X Axis.

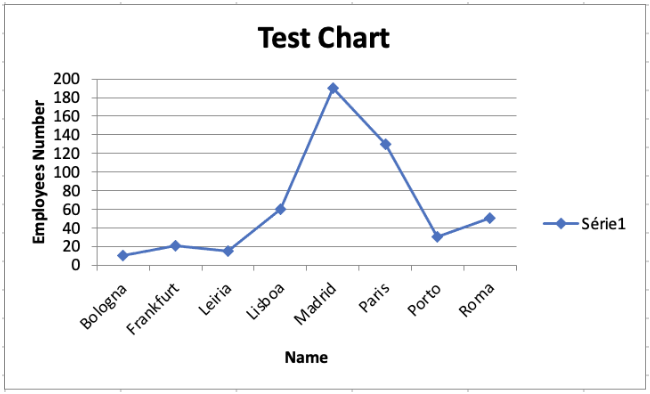


Fig. 7. Example of a chart

7 IMPLEMENTATION

In this section I will give a small explanation of how I did the some of the most important parts of the code.

When the button Import All Repository is used the VBA Module1 is called, the VBA starts by getting the inputs for the Python script from the cells that the user is expected to insert the inputs, then it is called the login pop-up asking for the username and password to authenticate in ATLAS API, after that the VBA will call the importRepository.py script.

The importRepository.py script is prepared to get all the classes and properties or just some, depending of a flag it receives. It starts by creating 3 Python classes the Class, Object and Property, used to save the data that is being imported from the ATLAS API, it is then configured a second Excel workbook in which it will be printed the data from the database. Then the script will iterate all the objects of the database and save the data about them in the Python classes and write it in the workbook just created using the pip library xlsxwriter, after all the objects are iterated it is saved the workbook and it is created a JSON file to be used later in the update database script.

Still in the VBA Module1, after the Python script is run, the VBA will read all the sheets in the workbook created and will copy them to the workbook where the code is being run so the user can then see them.

The Update Database button will call the VBA Module3, it starts by getting the inputs expected to run, it will then call the updateDatabase.py script.

The updateDatabase.py script starts by creating a request queue, which is a class that stores every request for the API to do at the end of the script, if there are no errors in the user alterations, it iterates the backup workbook and the JSON file generated in the importRepository.py script and compares with the sheets the user altered, it will then be verified if everything the user edited is valid, if the type of variables are correct and there are no errors, it adds the request to the queue. At the end if there were no errors it will run all the API requests from the request queue and will output two options, one where there was no problems with the request and another where one or more request had a problem to run.

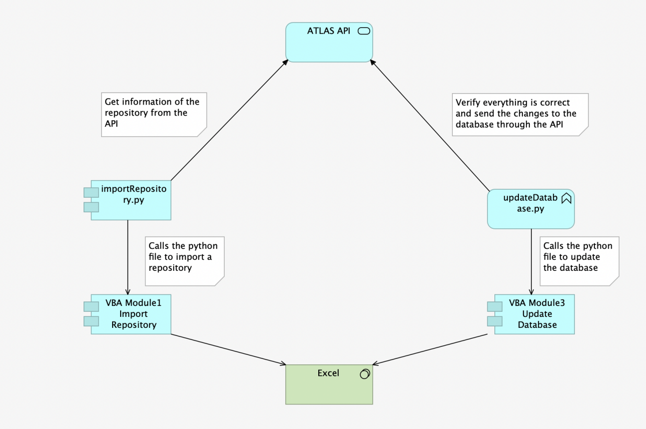

ATLAS API
Get information of the repository from the API
Verify everything is correct and send the changes to the database through the API
importRepository.py
updateDatabase.py
Calls the python file to import a repository
Calls the python file to update the database
VBA Module1 Import Repository
VBA Module3 Update Database
Excel

Fig. 8. Implementation Diagram

8 IMPLEMENTATION ISSUES

Here I will explain some of the issues for which no solution ended being developed and propose some solutions.

8.1 Concurrency Problems

The solution has some concurrency problems, if an user imports some classes and while he is editing the data, some other user edits the database, there can happen some conflicts, from the tests I made, the conflicts will not have any fatal crashes or nothing unexpected will happen to the database or to the Excel System, some cases that the requests will give some errors, which are warnings the request went wrong, are when the user edits an object that a concurrent user has erased, when the user adds an object with a name that is already being used by an object concurrently created he did not know about, or editing a property that was erased by a concurrent user or when an user references an object that was erased, in this case the system will create a new object to reference, but all the properties of the object the user wanted to reference will be lost, because the object was erased.

To solve this problem I came up with three solutions all with their trade-offs, one would be to have a listener that every time the user edits a cell would automatically update the database with that change, this would solve this problem, it would be only needed to see if there were changes in that value and if not update it, the problem in this solution is the performance would be really bad, because every time the VBA calls the Python scripts the Excel does not let the user edit anything until it is done. Another solution would be to have a lock mechanism that would not permit anyone else change the repository or even just some classes only while the user was editing them, this would also solve the problem but it is not ideal because we are blocking other users from editing the data. The last solution is to do the edition normally, and then before updating, iterating the ATLAS API log (a log which has the changes done in the database by the API) to check if there was any conflict in what the user edited, this will make the performance a little worst which is already not very good, but in my opinion is the best solution to this problem.

8.2 Performance Problems

I will show now some performance tests I did to the system, to test how much time the system takes to do the actions, here in the first table we can see how much time it takes to import some classes, it is important to notice this classes were created to test this and are all similiar with 50 objects with just a name property.

Table 1. Import Performance Tests Table

Test Detail	Time
Import 1 Class	4,44s
Import 2 Classes	4,71s
Import 3 Classes	4,79s
Import 4 Classes	5,15s
Import 5 Classes	5,38s
Import 10 Classes	7,67s

It is possible to see that to import a class will take around 4 seconds and then if we want to import more at the same time it will take around 0.3 seconds for each extra class, of course this values are variable and change from test to test, but always around these values. Now to show some updates to the database we can see a table of editing some properties using these classes used to test the imports before. In every column it's shown the result of editing properties in more classes, in the first it will update this properties for one class, than two classes and then five classes.

Table 2. Alter Properties Performance Tests Table

Test Detail	Time with 1 Class	Time with 2 Classes	Time with 5 Classes
Alter 1 Property	6,77s	7,11s	8,93s
Alter 2 Properties	7,06s	8,09s	11,05s
Alter 3 Properties	7,18s	8,54s	13,31s
Alter 5 Properties	8,25s	9,87s	17,32s
Alter 10 Properties	9,62s	14,14s	27,05s
Alter 30 Properties	18,95s	29,82s	1m 9,25s

We can see that it takes at least 6 to 7 seconds to update a property, that is because the system will verify everything is correct, will update the database and then it will get the data again from the database to make sure everything is correct, so as it takes around 4 seconds as we can see in the first table, it takes an extra 2 seconds to verify and update. Every extra property takes again around 0.3 seconds and when it is edited more classes it will take longer, but I would say the results are pretty linear, to the time it takes compared to the amount of properties edited.

The performance needs to be improved, because taking 7 seconds to import 10 classes and taking 1 minute and 9 seconds to alter 150 properties, I do believe is too much time, to solve this, there are some things that can be done, one is optimizing the API calls to the system so they take less time when called, another could be changing the programming language from Python to a more efficient programming language like C# or Java, another thing would be to try to make the code more efficient, even though I have worked on this already.

9 EFFICIENCY OF THE EXCEL SYSTEM VS ATLAS WEB APPLICATION

Table 3. Efficiency Tests Table

Test Details	Excel ATLAS Time	ATLAS Web App Time
Alter 6 properties in the same object	36s	40s
Alter 3 property of 8 different objects of the same class	55s	1m56s
Alter 3 property of 3 objects of different classes	1m3s	1m32s
Create 6 new objects of the same class	13s	31s
Create 2 new property, one of type text, and another of type reference	26s	37s

This times are taken from the time we need to spend in the website editing to get this changes done, and the time it takes to do the same in the Excel, I did not ended up running the requests in the Excel because as we can see in the performance tests, the amount of time it takes depends in the amount of data we have imported, and these results would depend a lot in the performance of the computer and the Internet used. And we can say that for small alterations, it is faster to use the ATLAS website but if the idea is to edit a lot of objects and in different classes it would be definitely faster to use the Excel system, even if then it takes a while to run the requests.

10 PROS AND CONS OF EXCEL

There are some pros and cons from using Excel as a tool to implement this system, as advantages the user has a really easy learning process because Excel has been used for so much time, almost everyone is familiar with their interface, it has a lot of features implemented that are very good to edit data and create charts, tables of the data. As disadvantages, Excel does not have the most common usage to make a system in them, so the fact that it gives so much freedom to the user can be a problem if they edit something that will then create bugs.

Advantages:

- **Very known and commonly used program**: Microsoft Excel has existed since 1985, most computer users are familiar with Excel.

- **Intuitive Interface**: As Excel is such a known program, it is really easy to use it, even if you as user do not know how to do something, there is always a lot of tutorials online, so everything becomes very intuitive.
- **Easy to do charts from the data**: Excel has features to create charts from the data in the sheets, the user selects a type of chart and then chooses the cells with the data for the chart.
- **Use formulas to analyse data**: The formulas Excel has, helps the user do some mathematical calculations, those calculations can be useful to make statistics of the data, getting average numbers, max, min and many other formulas.
- **Other features that make it easy to edit data**: Excel has also very interesting features to edit data, it has a suggestion system that can help rewrite values that are similar or alike, there is also the drag option to copy, where it can detect patterns and will continue the pattern for the cells you dragged.

Disadvantages:

- **Gives a lot of freedom to the user**: Normally, giving freedom to the user is a good thing, but in our case, since I built a system in which some cells cannot be edited without the system having some bugs, it is a problem for me that the user can edit everything.
- **Users need to know how to use the program, need to read the user manual**: A first time user maybe would have some trouble understating how the inputs and buttons work, that is why I created a user manual.
- **The user needs to have Excel and Python installed**: With a web application, there is no need to have anything installed on the computer, but with this solution the user must have Excel and Python installed to use it.
- **Does not have a very intuitive design**: It is difficult to create a intuitive and attractable design in an Excel sheet, so the design is very simple but very practical also in my opinion.

11 FUTURE WORK

For future work it would be really import to solve the Implementation Issues, choosing a solution with the trade-offs for the concurrency problem, and trying to find a solution to the Performance problems, one solution that would make the code a little faster would be to do optimized API requests just for this system, because there are a lot of API requests that get extra information that is not necessary for the system and make the requests slower, and even some requests that it's necessary to do class by class and it would be much faster to do one request for all the classes.

Here is a list of some improvements the system should receive for a better experience:

- **Making a design for the Control Panel sheet**: At the moment, the design of the Control Panel is very simple, but Excel is capable of doing some interesting designs with buttons, images, and more, it would be nice to have a more interactive design.
- **Restricting the cells the user can edit, for safety of the program**: To ensure that the program runs without errors, it would be good to lock all the cells that are not supposed to be edited in the Control Panel, because otherwise the user can create some bugs, this problem is mitigated by the fact that the users are all Link Consulting employees, so they are qualified to use the program.
- **Doing more efficient API calls specific for this program**: As I mentioned earlier, the API calls are not optimized for this program, which means that API sometimes gets too much unnecessary information, there are some API calls that need to be done class by class, and it would be much faster if there were a call for all classes. This change would be critical to improve the performance of the code, because API calls are the part of the code that takes the most time.
- **Deleting objects**: For better usability of the program, it would be nice to have an option to delete objects, it could be done by erasing the whole row of that object or with a special string to indicate to erase the object.
- **Deleting properties**: Deleting properties would also be very good, so the user does not need to go to the web app to do so, it could be done by erasing the column with that property or having a special string again to erase the column.
- **Create classes**: Another feature that could be added in the future would be to create classes by creating new sheets in the Excel workbook, the name of the class would be the name of the sheet, to create a class the only required parameter is the name.
- **Delete classes**: To delete classes, the user would erase the sheet of that class, but it would be better to ask for confirmation, because the user could do it by accident and losing an entire class by accident would not be good.

12 CONCLUSION

A system to edit a database with a tabular view is a very helpful tool, it allows us to edit the data faster than using the ATLAS website, even if the performance of the system it's not the best, the time it spares editing in the Excel versus the time spent in the ATLAS website going class by class would compensate the time we need then to wait for the system to update. And of course the fact that we can analyze the data using formulas, tables and charts using the Excel features helps a lot to using this Excel system.

ACKNOWLEDGMENTS

I would like to thank my supervisor Pedro Sousa and the employees of Link Consulting which helped me develop this thesis.

REFERENCES

ATLAS 2022. *ATLAS - Link Consulting.* Retrieved October, 2022 from https://linkconsulting.com/what-we-do/products/atlas/#CTA

DbVisualizer 2022. *SQL Client and Editor - DBVisualizer.* Retrieved October, 2022 from https://www.dbvis.com/

Object-Oriented Database 2022. *What is an Object-Oriented Database? | MongoDB.* Retrieved October, 2022 from https://www.mongodb.com/databases/what-is-an-object-oriented-database

Postico 2022. *Postico 2.* Retrieved October, 2022 from https://eggerapps.at/postico2/

Relational Database 2022. *What is a Relation Database?* Retrieved October, 2022 from https://www.techtarget.com/searchdatamanagement/definition/

relational-database

SQLGate 2022. *SQLGate - The Most Intelligent IDE for Database.* Retrieved October, 2022 from https://www.sqlgate.com/

Superset 2022. *Creating Your First Dashboard | Superset.* Retrieved October, 2022 from https://superset.apache.org/docs/creating-charts-dashboards/creating-your-first-dashboard/#creating-charts-in-explore-view