CUSTOMIZABLE INTERFACE FOR LTL CONFORMANCE CHECKING

The final document

Teodora Staneva
Mohamed Amine Kooli
Ugo Detaille
Vishisht Choudhary
Daniel Henel

Process Conformance Checking in Python
(WS 2022/2023)

Aachen, January 2023

Table of Contents

Project description 2

Used technologies and pre installation requirements 2

Installation guide 3

Detailed interface description 5

Use cases 8

Customer requirements 8

How does it work? 11

Research and analysis 11

Testing 12

- "BPI Challenge 2014" data set 12
- "BPI Challenge 2017" data set 13
- "BPI Challenge 2015" data set 13

Quality of the results 14

Team Members Reviews 15

- Ugo Detaille 15
- Teodora Staneva 15
- Mohammed Amine Kooli 15
- Vishisht Choudhary 16
- Daniel Henel 16

Other documents 17

Contact us 17

Project description

The aim of this software project was to implement an LTL Filter interface for event logs which can then be further used in different process mining use cases. The output of this application aims to take a first step in data cleaning and/ or preprocessing. The main objective of the project was to create a Web User interface with which the customer could interact, and which would then call upon the logical python units which were in the backend of the application.

The filtering functionality of our application was implemented in python as mentioned before. Based on our research we coded functions to create compositions of the following three filters:

Eventual follows filter is a technique used in process mining that helps to identify patterns of behavior in event logs. It is used to identify dependencies between events, such as which events tend to occur immediately after or before other events, and to remove unnecessary or irrelevant events from the log. The goal of the eventual follows filter is to simplify the event log and make it easier to analyze, by removing events that do not have any significant impact on the process.

The four eyes principle is applied as a filter in event logs, by having two or more individuals review each event before it is added to the log. This ensures that only authorized events are included in the log, and that any errors or fraud are caught and corrected before they can have an impact on the process.

Attribute different persons finds the process executions in which an activity A is repeated by different people.

During implementation we faced a couple of obstacles such as figuring out how to combine filters so the query of the user is completely satisfied. Furthermore, another issue we faced was connecting the frontend of our application with our python logic modules. The biggest problem was that a lot of the frontend was implemented in JavaScript and there was no direct possibility to pass arguments between the layers of our application. In the end we were able to solve these issues by using some tools and tricks.

Our repository containing our code, research and other documents can be found on GitLab under the following link:

https://git.rwth-aachen.de/teodora.staneva99/customizable_interface_ltl_checking

Used technologies and pre installation requirements

To work on the project and create the final product we have used different softwares, libraries and programming languages. The following mentioned helped us design the wished project and will be listed in this part of the document: To create the interface we used HTML, CSS and Java Script. HTML was used to create a structure for the Web User interface, CSS was for the formatting and design and Java Script for the dynamic and interactive content.

We also used Jupyter Notebook - an open-source web application that helps create and share documents that integrate code and other multimedia resources, along with explanatory text in a single document.

As to work on the backend we used the programming language Python and some of its libraries. Python provides great libraries to deals with data science application and for data manipulation. Some libraries that were used are Flask - web application framework written in Python; pm4py - Python library implementing a variety of process mining algorithms; Simpy - process-based discrete-event simulation framework based on standard Python.

Our project is stored as a GitLab project and can be accessed through:

https://git.rwth-aachen.de/teodora.staneva99/customizable_interface_ltl_checking

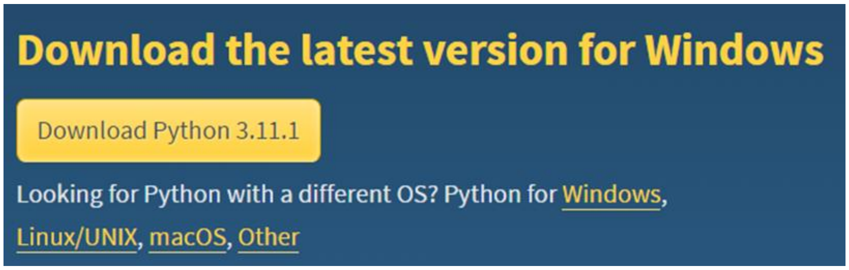
To help visualize the project and build, test, and deploy the application quickly we used Docker. Using Docker, applications can be deployed and scaled into any environment code will run without complications.

All of the mentioned technologies that need installation will be explained in the next section Installation guide.

Installation guide

❖ How to install python :

1. Go to the python’s download page and choose the OS you are using on your machine



Download the latest version for Windows

Looking for Python with a different OS? Python for Windows, Linux/UNIX, macOS, Other

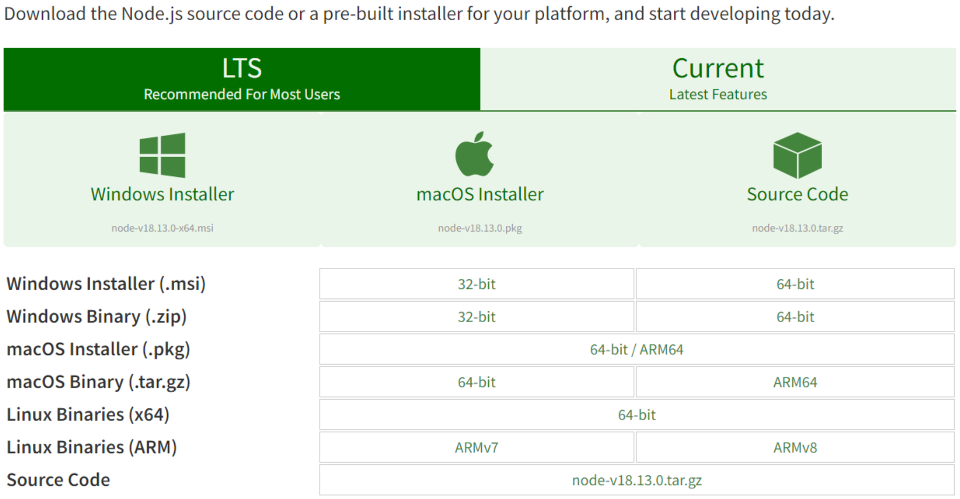
2. Go to the Python 3.10.0 version and download the installer.
3. Click on the downloaded file and follow the instructions on the installation wizard to authorize the installer and proceed with the install.

❖ How to install the libraries:

- Flask: open a terminal and type **pip install flask**
- Pm4py: open a terminal and type **pip install pm4py**
- Simpy: open a terminal and type **pip install simpy**

❖ How to install js:

1. Go to Nodejs download page and download the installer depending on your OS.



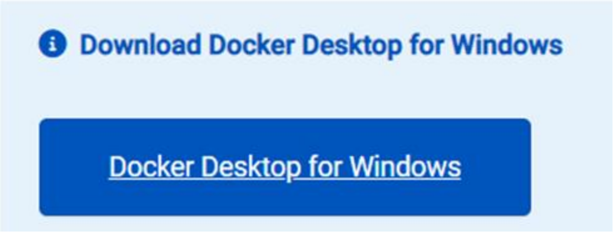
Download the Node.js source code or a pre-built installer for your platform, and start developing today.

Windows Installer (.msi)	32-bit	64-bit
Windows Binary (.zip)	32-bit	64-bit
macOS Installer (.pkg)	64-bit / ARM64	
macOS Binary (.tar.gz)	64-bit	ARM64
Linux Binaries (x64)	64-bit	
Linux Binaries (ARM)	ARMv7	ARMv8
Source Code	node-v18.13.0.tar.gz	

2. Click on the downloaded file and follow the instructions on the installation wizard to authorize the installer and proceed with the install.

❖ How to install docker:

1.Open the docker installation guide then download and run the installer by clicking on **Docker Desktop Installer.exe**



Download Docker Desktop for Windows

2. When prompted, ensure the **Use WSL 2 instead of Hyper-V** option on the Configuration page is selected or not depending on your choice of backend.

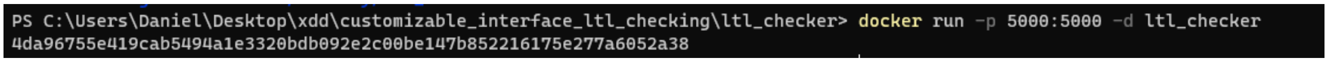
If your system only supports one of the two options, you will not be able to select which backend to use.

For more details see the system requirements section

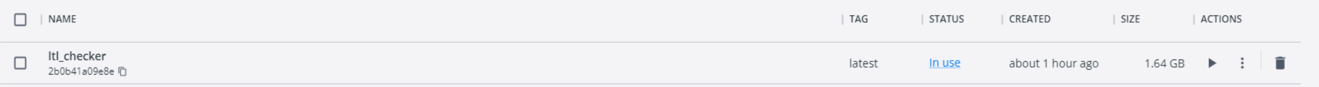
3. Follow the instructions on the installation wizard to authorize the installer and proceed with the install.
4. When the installation is successful, click **Close** to complete the installation process.
5. If your admin account is different to your user account, you must add the user to the **docker-users** group. Run **Computer Management** as an **administrator** and navigate to **Local Users and Groups** > **Groups** > **docker-users**. Right-click to add the user to the group. Log out and log back in for the changes to take effect.

❖ How to run the application using docker:

- Install git and docker on your machine
- Clone the repository
  git clone https://git.rwth-aachen.de/teodora.staneva99/customizable_interface_ltl_checking.git
- Go to the ltl_checker directory
  cd customizable_interface_ltl_checking/ltl_checker
- Build a docker image
  docker build -t ltl_checker .
- Run the docker image
  docker run -p 5000:5000 -d ltl_checker
- Open the application in your browser
  localhost:5000



```
PS C:\Users\Daniel\Desktop\xdd\customizable_interface_ltl_checking\ltl_checker> docker run -p 5000:5000 -d ltl_checker
4da96755e419cab5494a1e3320bdb092e2c00be147b852216175e277a6052a38
```



□	NAME	TAG	STATUS	CREATED	SIZE	ACTIONS
□	ltl_checker 2b0b41a09e8e	latest	In use	about 1 hour ago	1.64 GB	▶ ⋮ 🗑

❖ **How to run the application without docker:**

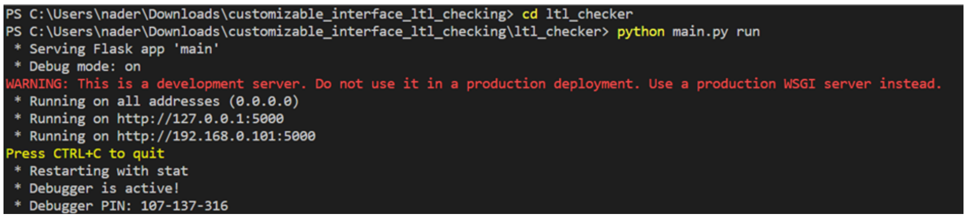
- Install git on your machine
- Clone the repository

  git clone https://git.rwth-aachen.de/teodora.staneva99/customizable_interface_ltl_checking.git
- Go to the ltl_checker directory

  cd customizable_interface_ltl_checking/ltl_checker
- Run the application

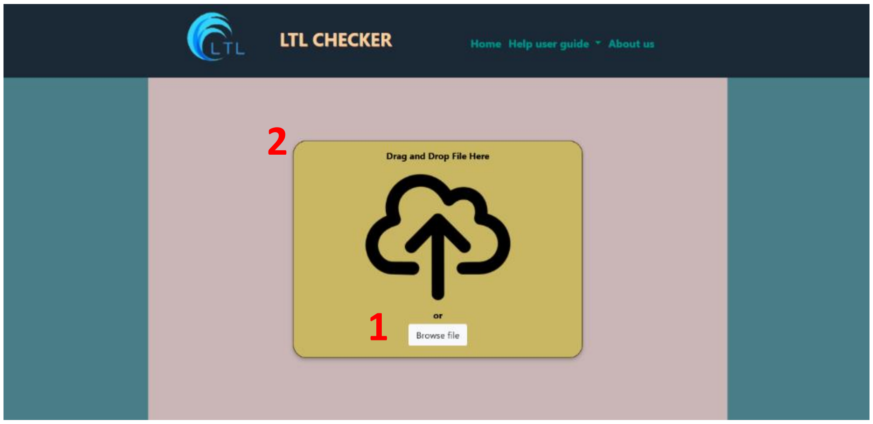
  python main.py run
- Open the application on your browser

  localhost:5000



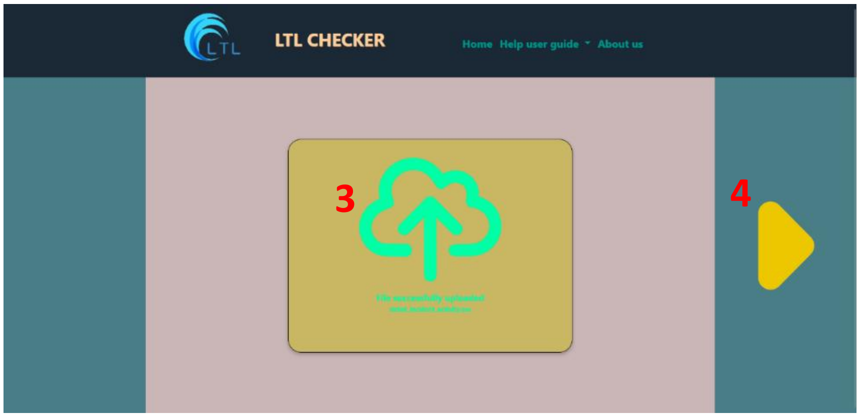
```
PS C:\Users\nader\Downloads\customizable_interface_ltl_checking> cd ltl_checker
PS C:\Users\nader\Downloads\customizable_interface_ltl_checking\ltl_checker> python main.py run
 * Serving Flask app 'main'
 * Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on all addresses (0.0.0.0)
 * Running on http://127.0.0.1:5000
 * Running on http://192.168.0.101:5000
Press CTRL+C to quit
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 107-137-316
```

Detailed interface description



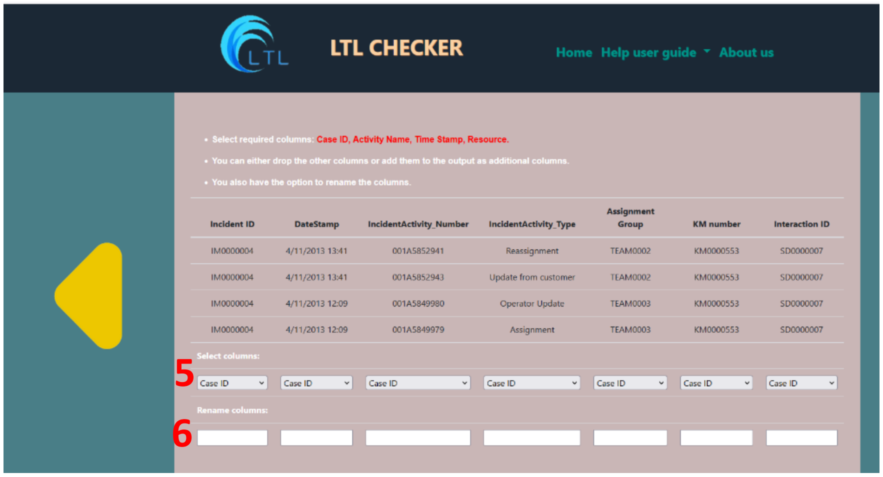
1 Browse File Button: **This Button opens the file explorer where the user will be able to select a file to upload**

2 Upload Square: **The user also has the option to drag the desired file onto the yellow field and upload the file into the application.**



3 Confirmation: If the Upload is successful the Icon will turn Green.

4 Navigation Arrow: With the successful upload a yellow arrow will appear on the right of the page with which the user can navigate to the next step.



- Select required columns: Case ID, Activity Name, Time Stamp, Resource.
- You can either drop the other columns or add them to the output as additional columns.
- You also have the option to rename the columns.

Incident ID	DateStamp	IncidentActivity_Number	IncidentActivity_Type	Assignment Group	KM number	Interaction ID
IM0000004	4/11/2013 13:41	001A5852941	Reassignment	TEAM0002	KM0000553	SD0000007
IM0000004	4/11/2013 13:41	001A5852943	Update from customer	TEAM0002	KM0000553	SD0000007
IM0000004	4/11/2013 12:09	001A5849980	Operator Update	TEAM0003	KM0000553	SD0000007
IM0000004	4/11/2013 12:09	001A5849979	Assignment	TEAM0003	KM0000553	SD0000007

Select columns:

Case ID | Case ID | Case ID | Case ID | Case ID | Case ID | Case ID

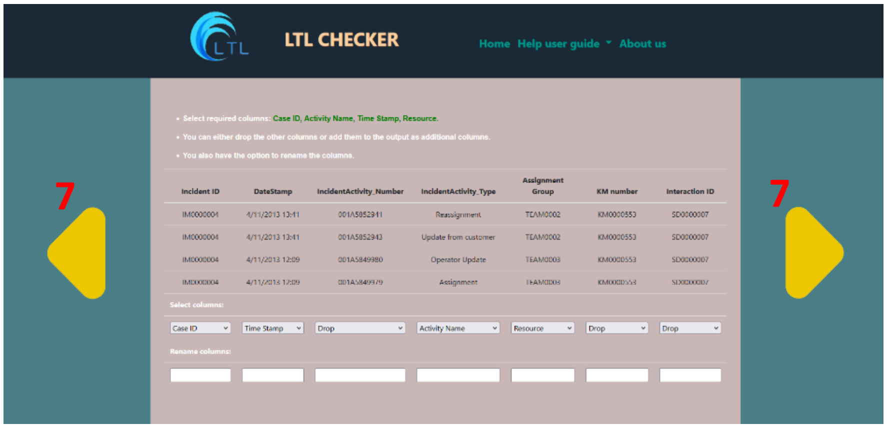
Rename columns:

5 Mandatory column selection

There is a set of mandatory columns which every event log must have. Using the pull down menus the user can create a mapping of the original event log names and the mandatory Column. In this case the original name "Incident ID" would be mapped to the mandatory name Case ID

6 Column renaming possibility:

Remaining columns can be arbitrarily renamed according to user preferences by typing the desired name into the text field.



- Select required columns: Case ID, Activity Name, Time Stamp, Resource.
- You can either drop the other columns or add them to the output as additional columns.
- You also have the option to rename the columns.

Incident ID	DateStamp	IncidentActivity_Number	IncidentActivity_Type	Assignment Group	KM number	Interaction ID
IM0000004	4/11/2013 13:41	001A5852941	Reassignment	TEAM0002	KM0000553	SD0000007
IM0000004	4/11/2013 13:41	001A5852943	Update from customer	TEAM0002	KM0000553	SD0000007
IM0000004	4/11/2013 12:09	001A5849980	Operator Update	TEAM0003	KM0000553	SD0000007
IM0000004	4/11/2013 12:09	001A5849979	Assignment	TEAM0003	KM0000553	SD0000007

Select columns:

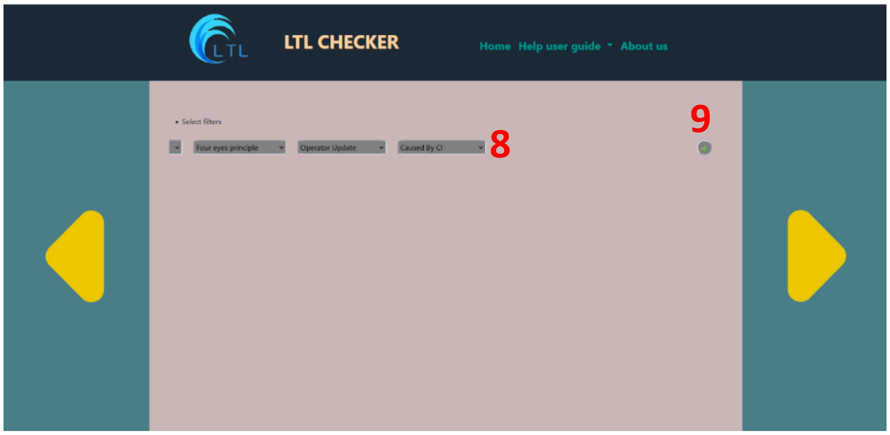
Case ID | Time Stamp | Drop | Activity Name | Resource | Drop | Drop

Rename columns:

Here we have an example of how the user might want to change the mappings of the column names.

7 Navigation arrows:

As mentioned before the user can click the arrows to navigate to the next or previous page



- Select filters

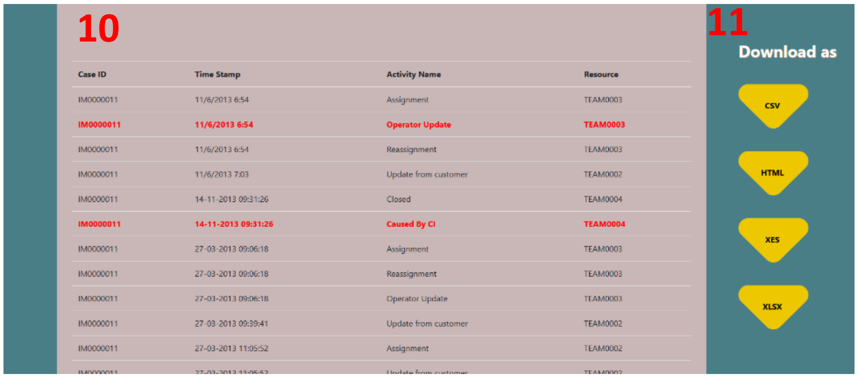
Four eyes principle | Operator Update | Caused By CI

8 Filter selection:

Using the dropdown menus, the user can input certain LTL features which we have mentioned in the beginning of this document.

9 Add filter:

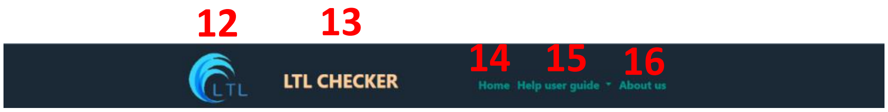
The "+" symbol enables the user to add more filters.



Case ID	Time Stamp	Activity Name	Resource
IM0000011	11/6/2013 6:54	Assignment	TEAM0003
IM0000011	**11/6/2013 6:54**	**Operator Update**	**TEAM0003**
IM0000011	11/6/2013 6:54	Reassignment	TEAM0003
IM0000011	11/6/2013 7:03	Update from customer	TEAM0002
IM0000011	14-11-2013 09:31:26	Closed	TEAM0004
IM0000011	**14-11-2013 09:31:26**	**Caused By CI**	**TEAM0004**
IM0000011	27-03-2013 09:06:18	Assignment	TEAM0003
IM0000011	27-03-2013 09:06:18	Reassignment	TEAM0003
IM0000011	27-03-2013 09:06:18	Operator Update	TEAM0003
IM0000011	27-03-2013 09:39:41	Update from customer	TEAM0002
IM0000011	27-03-2013 11:05:52	Assignment	TEAM0002
IM0000011	27-03-2013 11:05:52	Update from customer	TEAM0002

10 Filtered event Log preview:
After successful filtering a sample set of the filtered event log will be displayed for the user. The deviations will be marked in red.

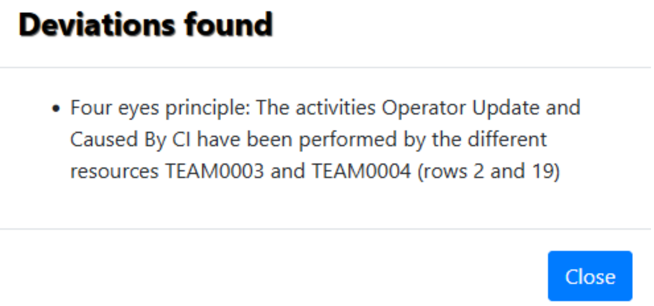
11 Download buttons:
Using these buttons it is possible to download the event log in one of the formats shown in the image (csv, html, xes, xlsx).



12 Logo, 13 "LTL Checker" and 14 Home:
By clicking the logo, the "LTL Checker" text or Home the User is navigated back to the first page where he/she can upload a file again.

15 Help User Guide:
After clicking this the user can view short explanations on how our interface works

16 About us:
By Clicking this the User is taken to a page with short descriptions of all the team members.



Deviations found

- Four eyes principle: The activities Operator Update and Caused By CI have been performed by the different resources TEAM0003 and TEAM0004 (rows 2 and 19)

Close

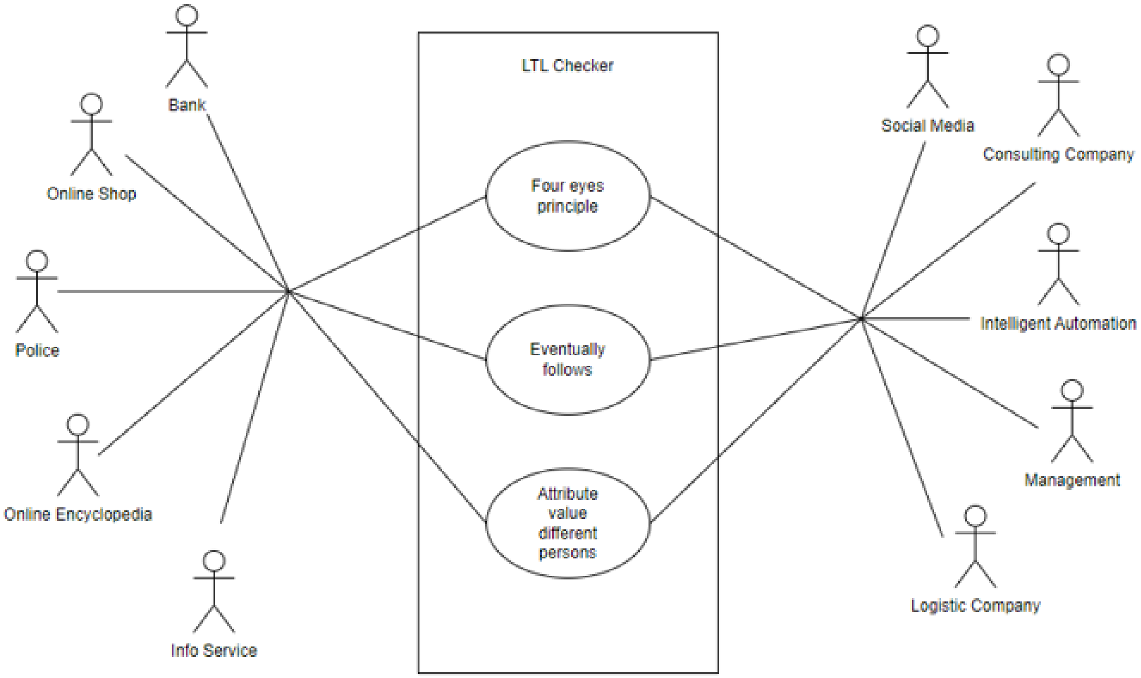
This Pop up displays the deviations found in the event log in natural language which makes them easier for the user to understand.

Use cases

Use cases for this field of data science are ubiquitous. In the following table we have given examples of how a bank could use our application in three different scenarios.

Case 1	When someone wants to get a loan, first the agreement must be signed by a customer and then approved by a bank employee, who must check if the customer has the permission to get the loan. We can provide **"Four eyes principle"** filtering to analyse log data in order to avoid such situations that employees approve their own credits.
Case 2	The bank's clients need to know statuses of their transactions, i.e. whether they are approved or rejected. We can use the "**Eventually follows"** filter to divide the transactions into these two groups. When a transaction is approved, the "Approved" activity must follow the "Start" activity. When a transaction is rejected, the "Rejected" activity must follow the "Start" activity.
Case 3	A client can be logged into his bank account on multiple devices simultaneously. Thanks to the **"Attribute value different persons"** filter we can detect insecure situations when the "Logged in" activity was performed on more than one device.

More examples you can find in our Project Initiation Document



Customer requirements

After completing our application, we could confidently state that practically all client requirements were satisfied. We have conducted our work so that we have always adhered exactly to the requirements of the engineering document that we had previously delivered. All requirements were stated clearly at the beginning, and we made an effort to fulfill each one. In particular, we took into account handling different event log file formats (such as csv and xes) and used the pm4py library to implement the filters' functionalities. Furthermore, the composition of various filters as combinations of activities and logical-temporal operations is considered and could be applied to different event log data. Thus, the deviations could be shown based on the process ID of the activities according to the used logical-temporal rules and the exact point of deviation in this process is identified. This enables us to assure the customer of a high-quality and reliable product. Additionally, the work was carried out on time with a well-organized and understandable documentation of the product.

The following table lists the hierarchically categorized requirements resulting from the requirements analysis process to show if each one of them is met in our final product:

Requirement	**Requirement type**	**met**	**Partially met**	**Not met**
The user should be able to upload event logs into the application	High-level	✓		
The user should be able to select different filtering types		✓		
The application should provide an intuitive and understandable interface		✓		
The user should have the option to save the filtered event logs to the device		✓		
The user should be able to select and combine features that need to be filtered (OR, AND)		✓		
The application should be able to show exact points of deviation with the possibility to explore the variants and the events (online conformance checking)		✓		
The user should be able to select certain features from the input data.		✓		
The code of the program should be understandable and readable		✓		
The documentation of the code should be thorough and precise		✓		
In case of an error the software should return an explanation of the origin of said error		✓		
The software should support various recent operating systems		✓		
The combinations of filters should give correct results according to LTL and logical operations.		✓		
The software should be optimized to run efficiently.		✓		
The user should be able to select required columns for the filtering process	Intermediate Level	✓		
The code should be modular and well commented			★	
The software should use different packages		✓		
The user should be able to select the format of the filtered event log.		✓		
The Program should run on Linux, Windows and MacOS		✓		
The functionality and performance of the code should be verified with extensive testing.		✓		
The Software should support Python 3.10.0		✓		
The library should provide methods for importing a process model and event log		✓		
The data in the originally uploaded event log should not be modified or changed		✓		
The implementation of the error handling in our application should be understandable to the user		✓		
The user should have the option to store the result as a PDF/CSV/XES/Parquet file.		✓		
The application view should be responsive		✓		
The user should be able to load event logs in CSV, XES or Parquet format into the application		✓		

The user should be able to select LTL filters (Four eyes principle, Attribute value different persons, eventually follows)		✓		
The library should re-use existing software where possible		✓		
The interface should be flexible and functional		✓		
Previously imported event log can be selected without importing it again	Low-level	✓		
The software should depend on PM4Py		✓		
The software should provide the possibility of grouping and sorting the results				✗
The software should show the results of different files in different tabs that can be used independently				✗
The user can choose language of the application (English, German + maybe our languages)				✗
The user should be able to filter the event log data by all features (activity, timeframe, ID etc.)		✓		
The user should be able to filter out events containing or not containing certain activities		✓		
The backend part of the application should use the PM4PY, Pandas and Numpy library		✓		
The graphical interface is to be implemented in flask		✓		
The test suite should simulate different python versions		✓		
There should be a manual on how to operate the software		✓		
The application should provide a list of all unique activities which can be used for filtering		✓		
The test suite should ensure that the implementation works correctly		✓		
The test suite should check for responsiveness and performance		✓		
The test suite should identify bugs and based on the information they will be fixed before the final product		✓		
The test suite will ensure the correctness of the generated model		✓		
The user can rename selected columns		✓		

To sum up, our product satisfied the majority of the requirements. Unfortunately, a limited number of low-level requirements could not be satisfied since they would require a lot of time and effort to develop and are not required for the functioning of the application's core features.

How does it work?

Our program works as follows: When entering the start page, the user encounters an upload window which he can upload the file into. The file shall be in CSV or XES format. Once the user has confirmed the upload, our program checks its compatibility. In any sort of error case, the cloud icon turns red and an error message appears, notifying the user that an error has occurred. Said error can have multiple reasons, but it should only appear if a wrong file format has been uploaded. Otherwise, the icon turns green and the user can skip forward to the next page by clicking on the yellow arrow that comes up on the right-hand side.

Now, the user has arrived at the main page: In a brief table format, the first few rows of the client's input are being displayed. In fact, the user can now modify his input: Firstly, the user may choose some column names he wishes to rename: Mandatory column names are indicated at the top: Those are **Case ID, Activity, Name, Time Stamp and Resource**. Other than that, the user may choose the columns' names to his preference and type the name into the text space below the columns. Secondly, the user can choose columns to be dropped from the table: Again, the mandatory columns cannot be left out, but all others can. For a column to be dropped, the user must click on the select icon and choose "DROP". Also, the user must choose the columns that must correspond to the **CASE ID, ACTIVITY NAME, TIME STAMP and RESOURCE** column. The selection of those columns works in the same fashion as when dropping a column.

Once the user has individually compiled his table, he can click on the right arrow again to in order to be led to the filtering page. Here, the user may select the classical process mining filters to be applied to his data set: **"EVENTUALLY FOLLOWS"; "FOUR EYES PRINCIPLE"; "ATTRIBUTE VALUE DIFFERENT PERSONS"** and so on. For each filter, the user must choose the columns that the rule shall be applied on. There are two buttons that determine the columns. The user must simply hover over those buttons and click on the conforming one: For example, the client may choose: "EVENTUALLY FOLLOWS"; "A"; "B" which would translate to: "Does A eventually follow B?" . Also, the client can combine single filters with one another by using logical operators. Those operators are "AND", "OR" and the brackets "(" and ")". So, the client can enter more complex inquiries as "Does A eventually follow B AND B eventually follow C?".

Before delivering the results, the program checks the input for correctness: In case any of the entered logical formulas may be wrong or some information missing, an error page (red icon) appears again and the client is asked to return to the prior page to correct the errors. Otherwise, the icon is green and the program works on calculating the result: Depending on the input size, that may take a few moments. Once all calculations are done, the result page appears displaying the results of the user's inquiries. The constraints that have not been matched are displayed in red in the wrongful rows. Also, a list in chronological order of all points of deviation is displayed below, highlighting why a certain row does not meet the users' request to the log. That file can be downloaded in multiple formats. The user can choose between CSV, XES and PDF format. Once the download is completed, the program is done.

Research and analysis

Since we were all new to the concept of process mining at the beginning of the project we had to do comprehensive research in order to create an optimal final product. Our main sources were from papers from process mining of which the titles can be found in our process initiation document (See 17.). Otherwise, we also got a lot of information from the PM4PY library which we included as a foundation for our project.

Jupyter notebook with basic filtering examples: basic_filters.ipynb

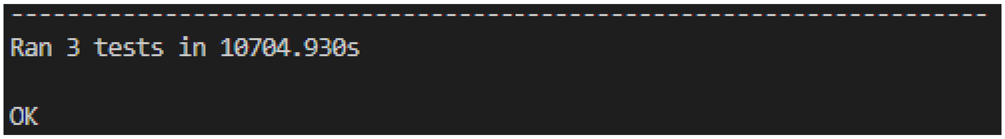
Jupyter notebook with LTL filtering examples: ltl_filters.ipynb

Jupyter notebook with explanation of our idea on combining the filters: filter_combinations.ipynb

Testing

We provide some performance tests on the selected function (**clac_result()** in the main.py file) that is essential because it calculates results of filtering. There were tests on performance that measured computation time depending on number of rows in the log file.

In our GitLab repository, you can find both the code with tests and the output in csv files, so you can take a look at the numbers.



```
----------------------------------------------------------------------
Ran 3 tests in 10704.930s

OK
```

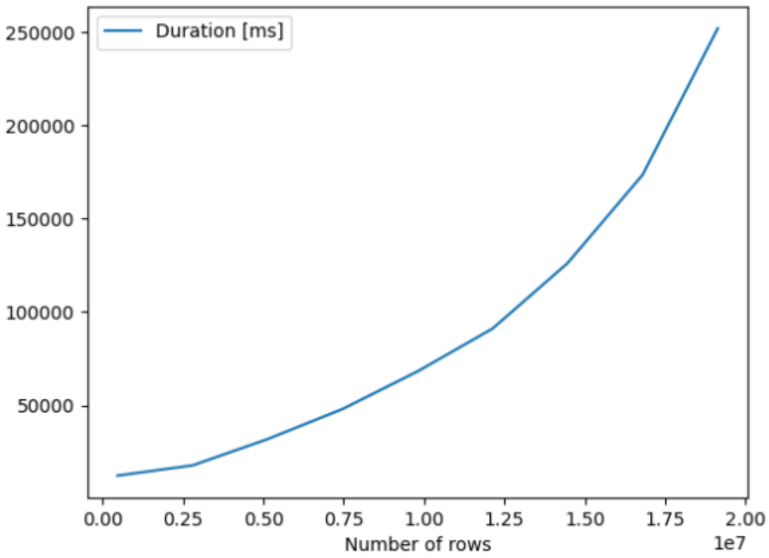
After almost **3 hours** of testing, there are following results:

"BPI Challenge 2014" data set

source: BPI_Challenge_2014 (4tu.nl)
type: big-sized logs
comment: exponential growth, but it still works enough fast (only 12s for almost 500k rows)

Number of rows	**Duration [ms]**
466737	12379.52
2800422	17851.06
5134107	31930.02
7467792	48067.53
9801477	68220.16
12135162	91152.38
14468847	126238.2
16802532	173539.8
19136217	251905.7

The whole output:
test-detail_incident_activity.csv

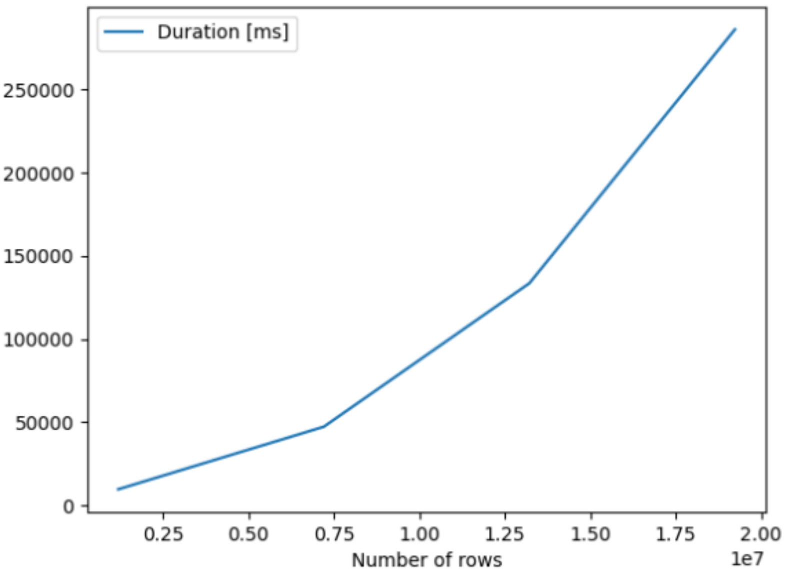


“BPI Challenge 2017” data set

source: BPI Challenge 2017 (4tu.nl)
type: enormous logs
comment: exponential growth, but it still works very fast (only 10s for almost 1.2M rows)

Number of rows	Duration [ms]
1202267	9695.753
7213602	47285.57
13224937	133513.3
19236272	286025.3

The whole output:
test-BPI_Challenge_2017.csv

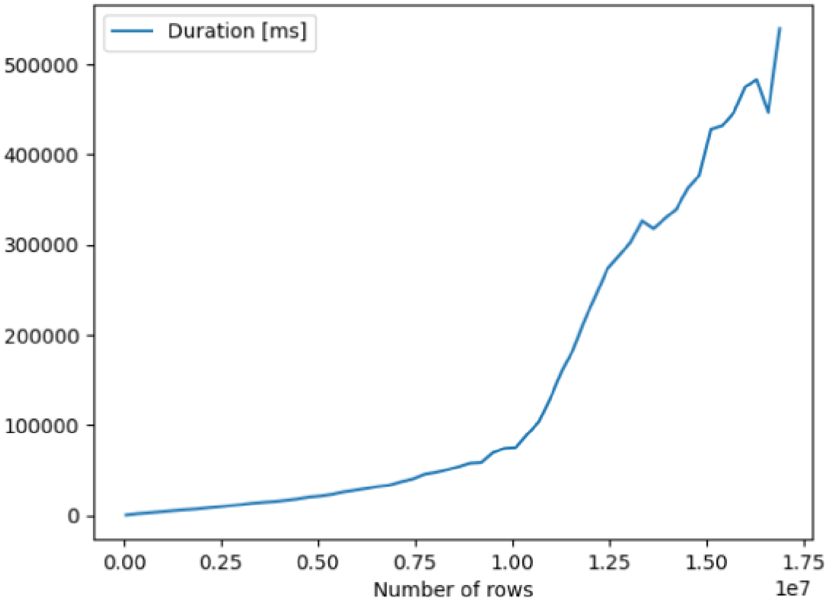


“BPI Challenge 2015” data set

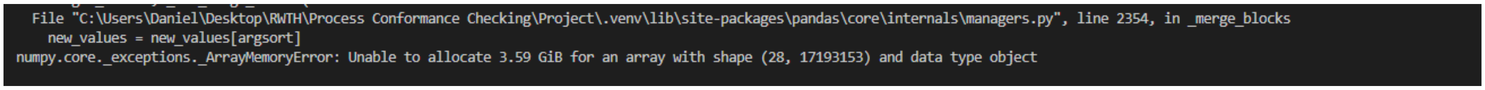
source: BPI Challenge 2015 (4tu.nl)
type: small logs, works very fast

Number of rows	Duration [ms]
59083	553.478
354498	1926.991
649913	2987.729
945328	4006.598
1240743	5133.618
1536158	6218.204
1831573	7020.305
2126988	8304.908
2422403	9473.986

The whole output:
test-BPIC15_5.csv



The resilience test failed for a dataframe with 28 columns and 17 193 153 rows.



```
  File "C:\Users\Daniel\Desktop\RWTH\Process Conformance Checking\Project\.venv\lib\site-packages\pandas\core\internals\managers.py", line 2354, in _merge_blocks
    new_values = new_values[argsort]
numpy.core._exceptions._ArrayMemoryError: Unable to allocate 3.59 GiB for an array with shape (28, 17193153) and data type object
```

There is only one factor against scalability – amount of memory that depends on the number of rows and columns! That's why it is so important to clean data and drop unnecessary columns before filtering!

Quality of the results

In this chapter we will present the quality results of our project.

Before the development phase we created a sophisticated plan describing all the requirements for a successful end product. Thanks to the planning steps we achieved our goal and delivered the finished product.

A part of an efficient project is scalability. It describes the ability of a software to perform well under an increased or expanding workload. In the case of our product an example is how well a database withstands growing numbers of queries. Extracting information from the presented above testing, we could see that the program holds well with increasing number of rows. It is usually accepted that a query becomes slower with additional data in the database. The growth in most cases is exponential – meaning more rows (more data to work with) more computing time. With every sprint that we worked on we have updated the front-end part of the project so that we that we can increase the usability of the software. Software deployment is a crucial step in achieving optimization, security, and compatibility. The end result is an user friendly and successfully working software compatible with multiple web browsers.

\

Team Members Reviews

Ugo Detaille

It has been a tough but also very interesting ride. First of all, I think we as a group can be really proud of the product we have created. It works really well and meets most of the requirements we had defined in the requirements engineering phase. I really like how self-explanatory our frontend is and how I believe the user may need only little to no explanation at all to understand the different steps of our program. I also really like the design and visualization of all icons and items on our webpage. The calculations are not very fast, the running time of our program is surely not ideal, but it still works fine on large(r)-scale inputs. But in fact, that field surely leaves room for improvement. Personally, I have learned a great deal about Process Mining, Python and all the frameworks we used: PM4PY; simpy and so on. From coding, to correcting bugs and reading through forums I think I really got a good first look at a field of data science.

I also think we did a good job evolving as a group over time. At the beginning, it was quite hard to deal with the different levels of knowledge and skills, and the organization was not perfect either. On a personal note, I really struggled with web page development, so I looked for tasks I was more prolific at (backend mainly : implementation of the functionalities in Python) But over time, we managed to improve on that as a group and became way more efficient and proficient because everyone got to find their niche they liked to work in. Finally, I am thankful to the team and Mr. Berti for offering that course.

Teodora Staneva

While choosing a project for the practicum I found this one really interesting and I'm glad I got assigned to it. The topic was fascinating, although I had no prior experience in programming with Python and no knowledge in process mining. I had to read a lot about the subject matter beforehand and I found that a bit challenging. Working on it was demanding and it took some time to get used to it, but thanks to great help from my teammates the project is now complete, and I think everyone learned a great deal. Most of the things I did were on the front-end side and I gained some new skills. Commutating with new people and working as a team helped me develope my communication skills even further – which will be very helpful in future work. In general, our meetings were successful, and we planned for our sprints and workload. I got to see and experience what working in the computer science field on such a project might look like. I learned a lot during these months, and I am happy with the results the team achieved.

Mohammed Amine Kooli

After completing the last steps of this project, I can say that I'm very proud of what my group and I have accomplished over the past few months. I clearly remember the first days when we were given the topic of creating a customizable interface for LTL Checking. I had no idea of this topic and how the end product would look like. So, I started learning more about it and dove deeper into the topic of process mining, which I found to be especially interesting because it's a part of the wide field of data science. I also believe that studying at the RWTH could be advantageous because of the strong chair of PADS and the excellent work they are doing. With time, I was able to enhance my coding skills, particularly on the front-end part, where it gave me great pleasure to work on the user interface and get results that were exactly what we had imagined when we started. I can say that my HTML, CSS, and JS coding skills have significantly improved, and I have gained knowledge of new frameworks like flask, PM4PY, SIMPY, etc. Additionally, I was able to learn how to create efficient unit tests for the implemented functions and how to resolve bugs.

Furthermore, this project gave me the opportunity to work in a scrum team for the first time and practice effectively assigning tasks to the team members. Trello and Git were other essential tools that helped us distribute the tasks more effectively and track how each of us is working through our individual tasks and what we have accomplished step by step. And with that, I feel I am fully prepared for the upcoming challenges working in a scrum team.

Although there were times when we were attempting to complete the majority of the tasks on the final day of the sprint, we were successful in delivering high-quality work each time, and I think that the ambiance and workflow were positive overall. Finally, I want to express my gratitude to everyone in my group for their work during this semester and to Mr. Berti and the PADS chair for offering this course.

Vishisht Choudhary

Having completed the project I must say that I am happy with our result. I think it was a very good opportunity learn about process mining using the PM4PY library. I was generally able to improve my python and data skills as well. I think for many people of the group this was the first time working as a team on code. One problem we therefore faced in the beginning was a knowledge gap between the teammates. We were able to bridge many of these gaps by efficiently communicating. Another problem was that in the beginning the work distribution in the team was a little bit off centered. But during later sprints we were able to manage distributing the tasks more evenly especially by efficiently using Trello.

As a result of this project I can say that Process Mining is definitely a field I can imagine pursuing later in my career. Through the contacts of the RWTH with Celonis and the PADS faculty I also think our University is a great place to start!

Daniel Henel

Before taking the project, I didn't have any experience with process mining, and I had to read a lot especially about LTL filtering. I just wanted to know what exactly process mining is and where it is used.
During our research we found a lot of real-life use cases and I must say I didn't know that this branch of data science is so common and so important in data analysis. That's very interesting that we can extract a lot of information from log data.

Since I am an Erasmus student, this was my first project with friends from the RWTH University.
At the beginning it was very challenging because I had to very quickly find myself in the new environment, but today I can say that I met great people and I really appreciate them for every moment I spent with them. We had to manage a lot of problems but there was nothing that could break the strength of our team because everyone could count on everyone in every difficult moment. We have learnt a lot from each other. I would like to mention not only the hard skills like programming and problem solving, but first of all the soft skills such as team working and task managing that I think are essential to create a project. Summarizing I think we have done a really good job and I am very proud of us.

Other documents

Most of the input, information and tools we used to create this project can be found in both or **Requirements Engineering** and **Project initiation Document**. For deeper explanation on the general topics, we may refer you to those two, where you can find our explanation and setup of the whole project. Still, we also offer you a shortened list of our sources down below:

https://thedigitalprojectmanager.com/tools/requirements-management-tools
https://www.jamasoftware.com/
https://www.modernrequirements.com/products/modern-requirements4devops/
https://thedigitalprojectmanager.com/tools/modernrequirements4devops-overview/
https://visuresolutions.com/home-new
https://www.orcanos.com/compliance/requirements-management/
https://cawemo.com/
https://www.botreetechnologies.com/blog/python-for-enterprise-app-development/
https://www.botreetechnologies.com/blog/top-10-python-use-cases-and-applications/
https://www.botreetechnologies.com/blog/flask-vs-django-comparison/
https://flask.palletsprojects.com/en/2.2.x/
https://www.tf-pm.org/resources/casestudy/analyzing-the-complains-prociess-at-granada-city-council
https://link.springer.com/chapter/10.1007/11575771_11
https://arxiv.org/abs/1905.06169
https://craft.co/jama-software

Contact us

Of course, there is still room for improvement in our project, as there is in any project. In case you may find bugs, errors, or functionalities that don't meet the requirements, please contact us via this e-mail: **amine.kooli@rwth-aachen.de**. **Amine Kooli** is our **Communication Manager** and Product owner, he will know how to deal with your problem best.