## CASE STUDY: Tickets per employee macro

CONTEXT

With no previous knowledge of VBA at all, this was the very first macro I had to build. Knowing the basic programming concepts from Javascript was a significant help to do this task.

PROBLEM

The ticketing system my company uses isn't capturing the amount of tickets that agents from a particular team worked on a particular day before they assigned it to a different team. In other words, the information that an agent from team X worked on a ticket is lost when the agent reassigns the ticket to team Y (and the original agent wouldn't be captured in the system's reporting).

However, when an agent works on a ticket, they put down a comment in form of a "Work note" before assigning it to another team. In the report then these "Work notes" would contain the information we need to find out how many tickets an agent worked on a particular day.

CHALLENGE

The system exports this daily report in an Excel format where the "Work notes" information is contained in a single column (*Pic 1*).

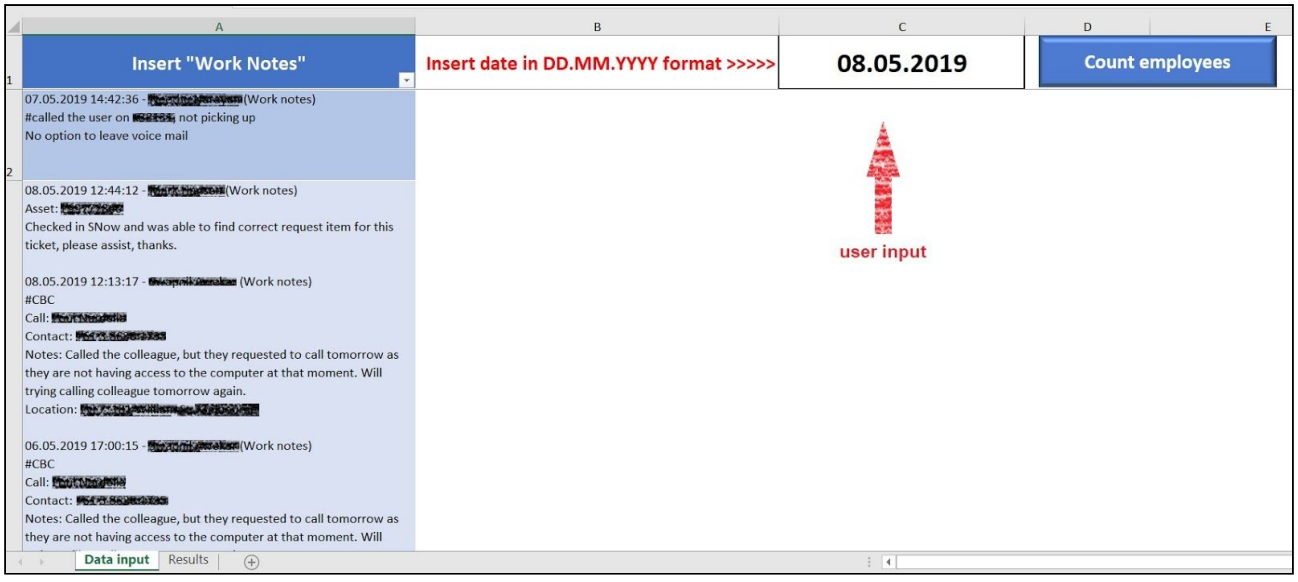
| | A | B | C | D | E | F |
|---|---|---|---|---|---|---|
| 1 | **Name** | **Reference Number** | **Created by** | **Short description** | **Work notes** | |
| 2 | sc_req_item | RITM0279679 | [redacted] | Install Fonts Locally on System | 07.05.2019 14:42:36 - [redacted] (Work notes)<br>#called the user on [redacted], not picking up<br>No option to leave voice mail | |
| | sc_req_item | RITM0282976 | [redacted] | Access to https://tableau.auiag.corp/#/site/IAG_Sandpit/projects/420/workbooks | 08.05.2019 12:44:12 - [redacted] (Work notes)<br>Asset: [redacted]<br>Checked in SNow and was able to find correct request item for this ticket, please assist, thanks.<br><br>08.05.2019 12:13:17 - [redacted] (Work notes) | |

*Pic no. 1 - The report showing "Work notes" for each ticket (one ticket = one row)*

Because these "Work notes" contain notes from each agent who worked on a particular ticket since it was open for the first time, I had to find a way how to extract the correct information and then find the final count of tickets per agent per day.

## SOLUTION

I decided to use RegEx to extract the date and employee name from the "Work note" and then store each "match" in an array. The date inside the Work note has to match the user input date (*Pic 2*)

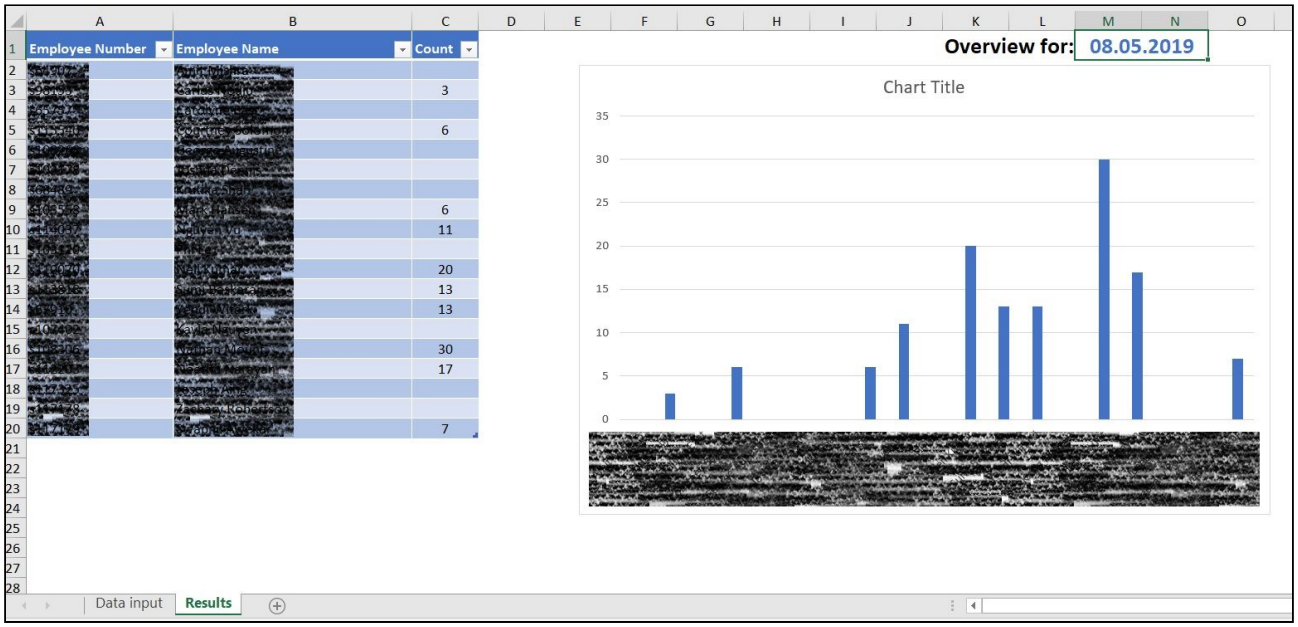



*Pic no. 2 - user pastes the "Work notes" into column A and inserts date they want to run the macro for*

Later, this array is compared with another array that contains employee names from a particular team and if match is found, an employee receives +1 to their total count.

See the code HERE.

After all tickets (all rows in our case) have been scanned, the total count represents the total amount of tickets an employee worked on that particular day (*Pic 3*)




*Pic no. 3 - result obtained after running the macro*

## RESULT

Since the report contains appx. 120–150 tickets per day, counting the tickets manually would be a long process with high chances of making a mistake.

By implementing automation of this task with macro, the team leader has a reliable tool to check daily performance of their team, now obtaining metrics that prior to automation weren't possible to get due to the complexity of the task.

## CASE STUDY: Creating new user ID macro

NOTE: this macro is part of larger automation that I'm currently developing and will add to the case studies later on.

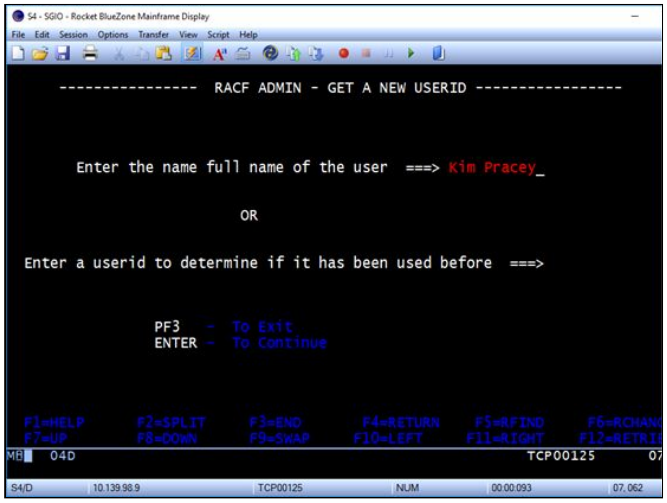
### PROBLEM

My colleagues provide support for various mainframe applications that we access via terminal emulator software. Navigating through these legacy systems and performing daily tasks such as adding, changing or removing information is highly user-unfriendly and extremely manual, repetitive and painful process.

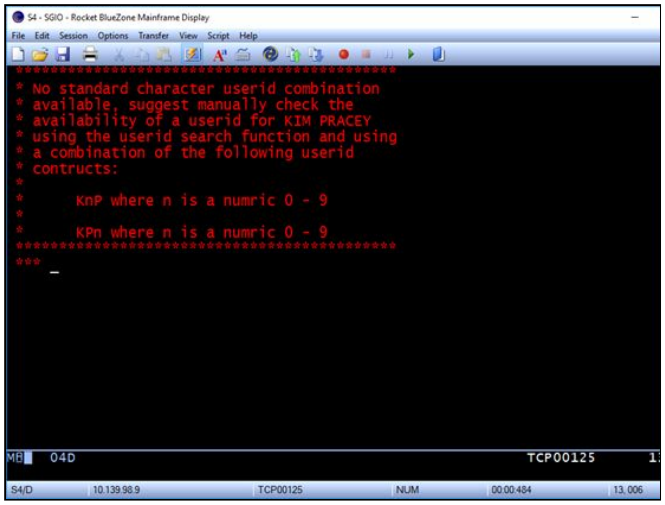
### CHALLENGE

Automate the process of ID creation for new users.

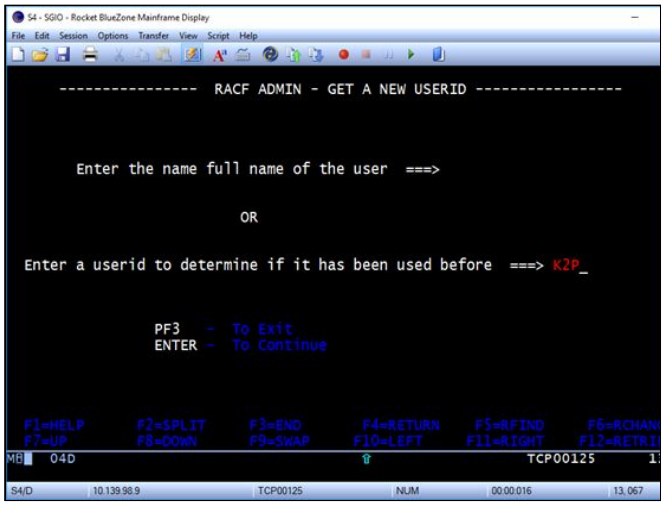
The biggest pain point here was to find an available user ID. After inserting user's name into the system, it would usually say there is no user ID available and the agent then had to go and manually try combinations of 3 characters (2 letters and 1 number) (*Pic 4 - 6*)




*Pic no. 4 - inserting user's name to find an available 3-character-long user ID*




*Pic no. 5 - the system advised us to manually find new user ID*

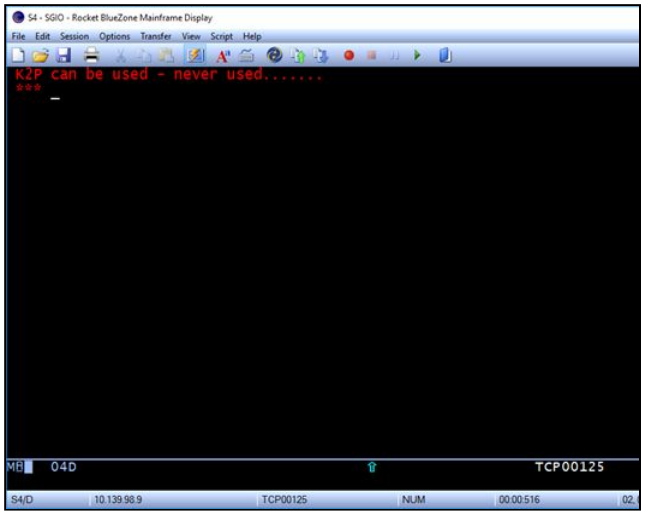



*Pic no. 6 - agent manually entering user ID variations*

## SOLUTION

I developed a function that scrapes the two letters “recommended” by the system and loops through their variation with a number 0 - 9 on the second or third position of the ID, inserting each variation into the system and scraping the output from the screen again.

Once the screen displays “can be used” (*Pic 7*), the subroutine exits the loop and saves the last ID variation into a variable and executes the next steps in the process of creating a new user ID.




*Pic no. 7 - screen after inserting a user ID that is available*

See the code HERE

RESULT

This macro is partially implemented, meaning we're using it for each new user ID request but I'm still fine-tuning the code according to the system behaviour. However, doing this task manually used to take appx. 10 minutes per user, depending on how quick the new user ID was found. Thanks to this macro the process now takes less than 1 minute per each new user.