Project Research Document

Employee Connectivity Tracker

X00149863 Pavel Ivanov

Bank of America project specifications:

- With the move to work from home in person interaction has been reduced between teammates. Implement an engagement tracker to see how teammates are communicating. Method could include Skype/Email/Webex/Phone/Mattermost/other.

- Allow an administrator to assign weighs to each of these different forms of communications and their frequency of use in order to identify teammates who may becoming isolated.

Detailed Discussion

In a discussion with BofA, we identified the products they are using for their day to day communications:

- Skype for its chat functionality.
  (Skype for Business will be replaced by Teams in July 2021)
- Mattermost for online conference calls.
- Outlook for email and scheduling meetings/events.

Assigning weight to communication apps

As per the specifications given by BofA, this system should allow assigning weights to each of the applications specified above: Teams, Outlook, Calendar.
The administrators should have the ability to see these apps and select the weight of each one in an intuitive way, such as a coloured sliding selector.

Grouping users in teams

Administrators should be able to create teams within the application and modify their members. A team would consist of multiple members, one of which would be the team lead that should receive all system notifications related to their teammates.

Data visualisations

All users will be able to view the communication data they have generated.
Whereas team leaders should not only have access to their own detailed data, but they should also be able to see the summarised communication data of their teammates.

All users that can view their data should be able to perform some sort of filtering, e.g., if they filter by date, they can view information generated:

- On that day (This should be the most detailed report out of all).
- For the current week.
- For the current month.
- For the past month.
- Since the start of the year.

Notifications for potentially isolated employees

The system should send out email notifications when a user's communication readings have significantly dropped in the past week compared to their average stats for the month before.

- These notifications should be generalised. The emails the recipients might receive would essentially state something such as "Hey! It seems *<user's name>* has not been in touch with other people for a while. You might want to check in on them".
    - The users that receive the email should not be aware of the actual stats of the user from the notification.
- The recipients should be configurable and can be either only the team leader, or the whole team.
    - The users allowed to manage these configurations should be the team leader and the administrators.

Users and access to functionality

Employee:

- Can only sign in with a Microsoft account (so that it be configured to allows users of a specific organisation, e.g., BofA or TU Dublin).
- Communication apps are tracked by Microsoft (Teams, Outlook and Calendar).
- Can receive notifications about isolated teammates (only if configured).
- Can view the data the tracker has recorded of their communication activities.
- Can filter the communication data by predefined date ranges.

Team lead (is also an employee):

- Will *always* receive notifications about isolated members of their team.
- Has access to a summarised view of the communication data generated by the other members of the team.

Administrator:

- Communication activities should not be tracked
    - It may also be possible to have separate login credentials instead logging in with a Microsoft account.
- Will receive system notifications only for specified team leaders.
- Can view communication data
    - For all employees summarised per team.
    - Can drill down on a specific team and see data for individual employees.
        - The data on this view should be anonymous.

Existing Applications in this domain

Applications that fit the requirement to record only the communication data generated by a person are not that common, well developed, or popular. However, there are a lot of software solutions that track all applications' usage on a computer.

Application	Similarities	Differences
ProcrastiTracker	• Tracks what applications the user is using and for how long. • Visual representation of the gathered data.	• Application runs natively on the machine being tracked. • Tracks all applications and documents. • Counts keystrokes and clicks.
TMetric	• Tracks what applications the user is using and for how long (only when the user starts a timer). • Visual representation of the gathered data.	• Application runs natively on the machine being tracked, allowing it to run offline. • Tracks all applications and documents. • Groups data by projects. • Request Time Off feature. • Can set billing rates and export recorded data as an invoice.
WhatPulse *(Application is not related - it only counts keystrokes)*	N/A	N/A

Platform, technologies, and libraries

The user will interact will a C# ASP.NET Core MVC application that will be hosted on Azure. The reasoning behind this decision is because Microsoft will take care of the authentication and the .NET framework was originally developed by Microsoft and has an authentication library (MSAL) which will be very useful in retrieving an access token for a user.

The access token can then be used to authorize the HTTP requests to Microsoft Graph API, which just happens to store most of the data required for this application. Most of the endpoints provide a lot of optional query parameters, such as count, filter, orderby, etc. Such parameters can be useful as the API's internal processing methods are most likely already optimised and efficient enough.
Once the data from Graph API is retrieved in JSON format, it can be filtered and stored in a relational database – Azure SQL Database.

Identified libraries at this stage:

- Microsoft Authentication Library (MSAL)

  Enables Azure Active Directory authentication, which in terms can provide access to Microsoft Graph API.
- EntityFrameworkCore.SqlServer

  Object-relational mapping framework/library - It will allow the program to access the Azure SQL Database (SQL Server) in an abstract manner.

The risks

In my opinion the biggest risk in this project can be the dependency on Microsoft Graph API. If for some reason the data flow from Graph API stops, the application will not function as most of the application's functionality is completely dependent on it. This is not likely to happen but if it did, that would be devastating.
This includes Graph API endpoints returning incorrect / unusable data, or a critical bug in the authentication library.

Another risk would be the user(s) not logging into the app frequently.
Since Microsoft will be authenticating the users, the application cannot store credentials and maintain a constant stream of incoming information.
At this stage, the plan is to extract the most recent data related to a user whenever they log in. If they do not, then notifications for isolated people will not be sent and any summarised data may be inaccurate.
There should be a workaround for this specific case.

Dependencies always have the potential of introducing unwanted behaviour or even break functionality. The identified dependencies are not that risky, but the possibility is still there.

The specifications are still a bit vague and this type of application will be a new thing for me. In this case time management can be a risk as well. It should be a problem, but it has the potential to be one.

Another potential problem would an unstable internet connection when working with cloud technologies such as Azure or Google Cloud.

No other risks have been identified at this point.