Project Transition Report

DriveMKB Quickscan

Max van Hattum

11 sep 2023

openmaze.

1. Table of Contents

2. Version history

Version	Date	Changes	Author
0.1	11 sep 2023	*Initial setup of the document, table of contents with all topics, introduction, project overview, current status, technologies used.*	Max van Hattum maxvanhattum @openmaze.io
0.2	12 sep 2023	*Dependencies, future steps, conclusion*	Max van Hattum maxvanhattum @openmaze.io
0.3	12 sep 2023	*Deployment overview*	Max van Hattum maxvanhattum @openmaze.io
0.4	12 sep 2023	*Process comments*	Max van Hattum maxvanhattum @openmaze.io
1.0.0	19 sep 2023	*Add deployment details*	Max van Hattum maxvanhattum @openmaze.io

3. Document distribution history

Version	Date	Recipients	Comments
0.3	12 sep 2023	*niekvandam@openmaze.io*	In line.

4. Introduction

This handover document outlines the journey of OpenMaze in developing the Quickscan, an initiative by the Centre of Expertise Interaction Design at Fontys, part of the DriveMKB project. The goal was to research and develop a tool that would empower small business owners, enabling them to gain rapid and effortless insight into their level of digitalization.

Throughout this document, key aspects of the project will be discussed, including its primary features, important technical details, such as the technologies utilized, deployment procedures, and content customization options. Additionally, potential future steps to further enhance the Quickscan tool will be explored, ensuring its continued relevance and effectiveness.

The main goal is to enable other parties or individuals to theoretically use this tool themselves, and customize it to a certain extent, while touching upon future possibilities.

5. Project overview

OpenMaze dedicated eight months to the development of the Quickscan tool as part of the DriveMKB initiative. The primary objective of the Quickscan is to empower small business owners with the data-driven insights and knowledge necessary to make informed decisions in the context of digitalization.

The goal was to create a user-friendly and efficient web application that fulfills several key criteria like:

- Ensuring ease of use
- Streamlining the process of gathering insights
- Automating data collection from online sources

Specifically the collection of data is vital as it enables business owners to gauge their digital presence comprehensively.

6. Current status and key features

6.1. Homepage and User Engagement

The Quickscan tool is designed to engage users from the moment they land on the homepage. The primary goal is to inform users about the tool's purpose and to encourage them to fill in the Quickscan. The homepage provides clear messaging, enticing visuals, and a compelling call to action, inviting users to begin their Quickscan.

6.2. Quickscan tool

The Quickscan can easily be started straight away from the homepage. Users are prompted to provide information, such as their business name, region, and Instagram username. Additionally, they are presented with a concise set of questions related to their digital presence and business activities. Note that these questions can be changed with little effort, as it has been set up in such a way that the questions will automatically be adjusted and saved in the database.

6.3. Data Retrieval from Google and Instagram

The tool collects data from various sources, including Google and the Instagram Graph API made by Meta.Google is used to gather business information such as:

- Their Google rating
- The number of Google reviews
- Available services (e.g., reservations, online ordering)
- Overall recognition on Google.

Instagram is used to gather information like:

- The number of posts and followers
- The mean number of likes and comments per post
- The post frequency per week

6.4. Survey Data Handling

In parallel with data retrieval, the Quickscan captures and organizes user survey responses. The responses can be yes, no or consciously chosen no. Responses are grouped and stored according to the survey questions themes, ensuring that user-generated data is structured and accessible for analysis. These answers are used to generate a score, where yes is good, no is bad and the other option is disregarded from the score calculation.

6.5. Customized Results Page

After completing the Quickscan, users are directed to a personalized results page of their business. On this page, on the top, a bar chart that displays the mean scores across all assessed themes is shown. This visual representation offers a quick overview of their digital performance relative to their peers.

6.6. Detailed Insights via Histograms

Scrolling further down the results page, users gain access to detailed insights through histograms. These histograms provide an in-depth look at their performance in individual subjects, utilizing a percentile-based scoring system to show where they stand in comparison to others in the same industry. For the survey section the advice part is dynamic. Every question has specific advice coupled with it, and is only shown if the user answered ‘No’ to said question.

The current status of the Quickscan project reflects a robust and user-focused tool that guides users through the assessment process and provides comprehensive insights into their digital presence. This combination of user engagement, data retrieval, survey handling, and presentation of results ensures that small business owners can make informed decisions to advance their digitalization strategies effectively.

7. Technical details and deployment

As of the current status, OpenMaze is responsible for hosting the Quickscan website. This section provides an overview of the technologies used, deployment requirements, deployment guidance, and customization options. This document refers to two code repositories, the *'quickscan-frontend'* and *'quickscan-backend'*.

It's important to note that while the majority of the content is hard-coded, there are dynamic elements, specifically the survey questions and the rendering of results and advice. These dynamic aspects are facilitated by the backend, and this section will delve into the specifics of adjusting these questions and will provide a high-level overview of the tool's adaptability for potential modifications.

7.1. Technologies Used

The software project consists of two parts, the frontend, which is the website the users see, and the backend, which is a REST api configured with a database. The frontend is built with NextJS a modern React Framework (*Next.js by Vercel - the React Framework*, n.d.). The backend is built with NestJS and the NoSQL database MongoDB (MongoDB, n.d.; *NestJS - a Progressive Node.js Framework*, n.d.).

Both parts are configured for Docker, a containerization technology, which makes deploying easier ("Docker: Accelerated Container Application Development," 2023). Because of this the application can be easily deployed to other servers, by simply executing the "build" and "up" commands of Docker Compose. The specific steps will be detailed below.

The application is lightweight, a relatively cheap Virtual Private Server (VPS) can be used when the user counts are low. They should, of course, be scaled upwards if web traffic is high. An example VPS specifications for little traffic is:

- RAM:2 GiB
- CPU: 2 vCPUs
- BANDWIDTH: 3.000 GiB
- SSD STORAGE:60 GiB

7.2. Dependencies

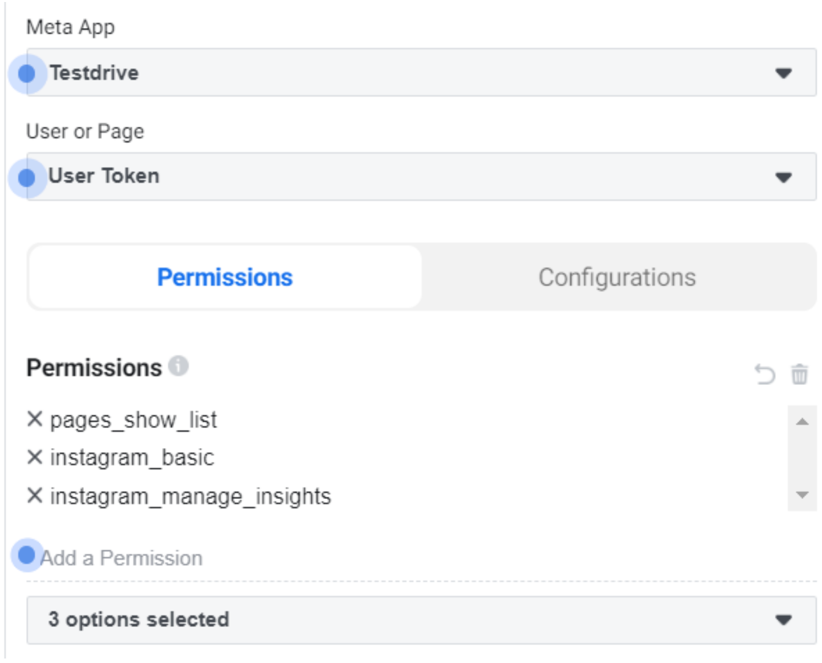
It is of importance to note that the application needs a Facebook Graph API key to get the Instagram data, and a Google API key to get data from the Google Places API.

7.2.1. Facebook Graph API

The application does not use Facebook login, but an Access token linked to a Facebook app page. There is a need for a never expiring API token, some steps should be taken so that this is ensured. Firstly, create a Business Facebook and Instagram and link them together. Secondly, using a developer facebook account, create a Facebook app. In the Graph API Explorer from Meta, it is possible to easily configure what is needed for the token. It must be a 'User' token, linked with the just created app, with the scopes:

- pages_show_list
- instagram_basic
- instagram_manage_insights

See below for an example of this.


Meta App: Testdrive
User or Page: User Token

Permissions
- pages_show_list
- instagram_basic
- instagram_manage_insights

3 options selected

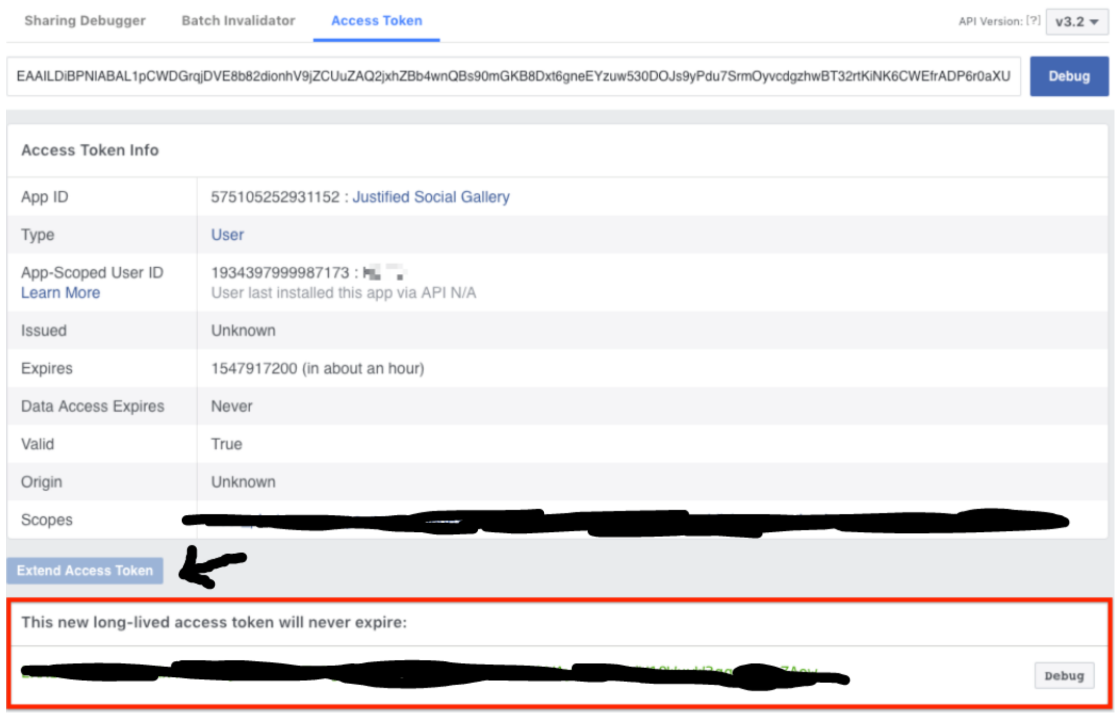
See below links to get to these resources.

App Dashboard

Graph API Explorer - Meta for Developers

Then, the token must be transformed to a never expiring token, which can be done in the Access Token Debug Tool from Facebook. Go to: https://developers.facebook.com/tools/debug/accesstoken/

And paste in your access token from the previous steps and press debug. Then press 'Extend Access Token'.



EAAILDiBPNIABAL1pCWDGrqjDVE8b82dionhV9jZCUuZAQ2jxhZBb4wnQBs90mGKB8Dxt6gneEYzuw530DOJs9yPdu7SrmOyvcdgzhwBT32rtKiNK6CWEfrADP6r0aXU

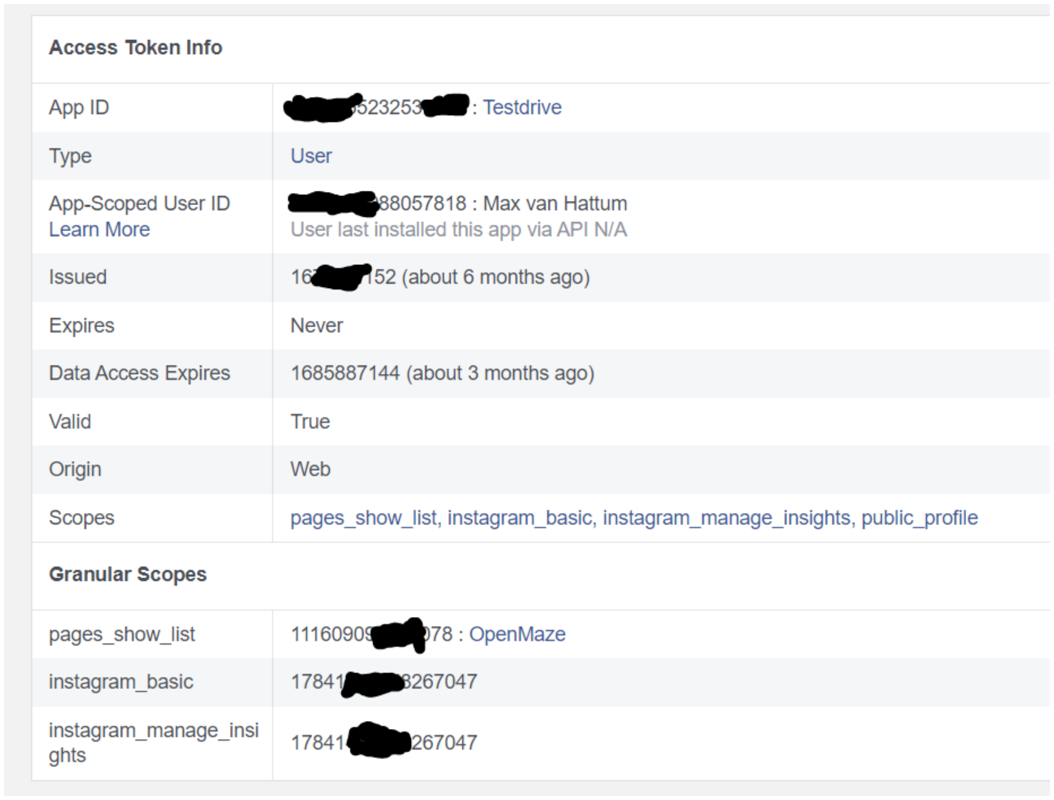
Access Token Info

App ID	575105252931152 : Justified Social Gallery
Type	User
App-Scoped User ID Learn More	1934397999987173 : [redacted] User last installed this app via API N/A
Issued	Unknown
Expires	1547917200 (in about an hour)
Data Access Expires	Never
Valid	True
Origin	Unknown
Scopes	[redacted]

This new long-lived access token will never expire:

[redacted]

Now the 'Expires' field should be set to 'Never'.



Access Token Info	
App ID	[redacted]523253[redacted] : Testdrive
Type	User
App-Scoped User ID Learn More	[redacted]88057818 : Max van Hattum User last installed this app via API N/A
Issued	16[redacted]52 (about 6 months ago)
Expires	Never
Data Access Expires	1685887144 (about 3 months ago)
Valid	True
Origin	Web
Scopes	pages_show_list, instagram_basic, instagram_manage_insights, public_profile
Granular Scopes	
pages_show_list	11160909[redacted]78 : OpenMaze
instagram_basic	17841[redacted]267047
instagram_manage_insights	17841[redacted]267047

Lastly, it is needed to get hold of the Facebook page Id and the Instagram Page Id. See below resources on how to get these.

Facebook id: [Getting Started - Instagram Platform]

Instagram id: [Getting Started - Instagram Platform]

Keep note of the access token and the Instagram id, as they will be needed when configuring the application. With this access token, the application is allowed to make 200 api calls per hour. Which for this phase of the project is very much sufficient. However, in a later stadium, Facebook login can be implemented, after going through the Facebook application process. This will up the limit with 200 extra requests per active user on the Quickscan.

7.2.2. Google Places API

The data from Google is accessed by using the Google Places API. Google has a guide on how to get and configure a key that is needed to make the calls.

Use API Keys with Places API | Google for Developers

7.3. Deployment and customization

Deployment is made easier because of the use of Docker, first a basic overview will be given, then specific steps will be shown to start up the complete application.

7.3.1. Overview

First, there are two Git repositories (quickscan-frontend, quickscan-backend) which must be cloned. Both the repositories make use of environment variables to configure the applications properly. Example values can be found in the repositories. This is also the place where the above-mentioned keys must be set. On top of this, the domains (URL's) where the applications are going to be accessible must be configured here.

Then, both applications can easily be built into Docker images using the accompanying Docker files. There are also docker-compose files present to easily start the MongoDB database with the backend.

This can easily be done on a VPS as mentioned above. Then a reverse proxy should be configured to expose the frontend, as well as the backend. Finally a domain can be configured to point to the VPS.

It is also possible to change some of the content. Textual content is mostly embedded in the frontend code. However, the questions for the survey can be easily changed *at the start* of configuring the backend. This is a simple matter of adjusting one file in the backend repository.

7.3.2. Repositories

There are two repositories, *"quickscan-frontend"* and *"quickscan-backend".* Both are needed for the application to function. We will refer to these repositories below as frontend and backend.

7.3.3. Environment variables

The frontend has a file '*env.local.example'*. Copy this file and name it '*env.local'*. Change the value of url to the domain where the backend will be hosted.

The backend has a file *'env.example'*. Copy this file and name it .env. Change the values of:

- GOOGLE_MAPS_API_KEY = *your Google Api Key*
- META_API_KEY = *your Facebook token mentioned earlier*
- INSTAGRAM_BUSINESS_ID = *your instagram id mentioned earlier*
- CORS_URLS = *the url of where the frontend will be hosted*

Make sure to have these files present in the folder when building/running

7.3.4. Content changes

Most of the text is embedded in the frontend repository. With a smart code editor it is possible to search through all the files for specific lines of text, and there they can be adjusted. The result page explanations however are stored in the backend, search for them there.

The survey questions and their advice are gathered in one file in the backend:
src/modules/survey/data/questions.data.ts

Make sure to observe the data structure and matching id's for questions and advice. Note, this must be done *before* building the backend image, which is going to be detailed below.

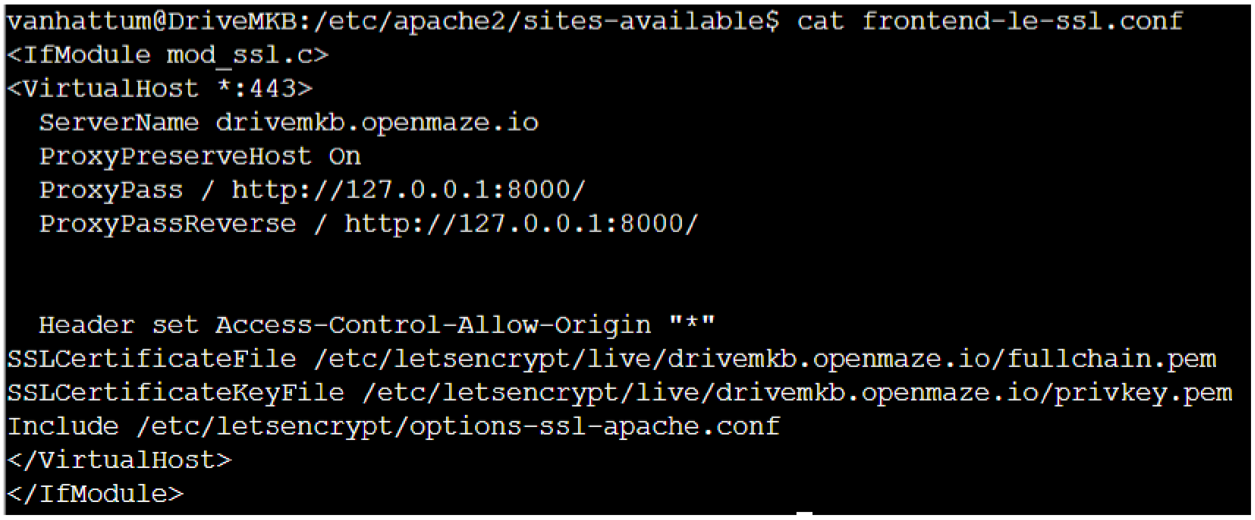
7.3.5. Docker

Docker compose can be used to build and start containers. The easiest way to deploy the application is to clone the repositories on the VPS, make sure Docker is installed. Then in both repositories, make sure the *".env"* files are present and correct. Finally just run the command *"docker-compose up –build -d"* in both repositories.

Note that it would be more performant to build those images somewhere else and push them to an image registry, Then they can be pulled from this registry to the VPS, and just started without building: *"docker-compose up -d".*

7.3.6. Example Reverse Proxy

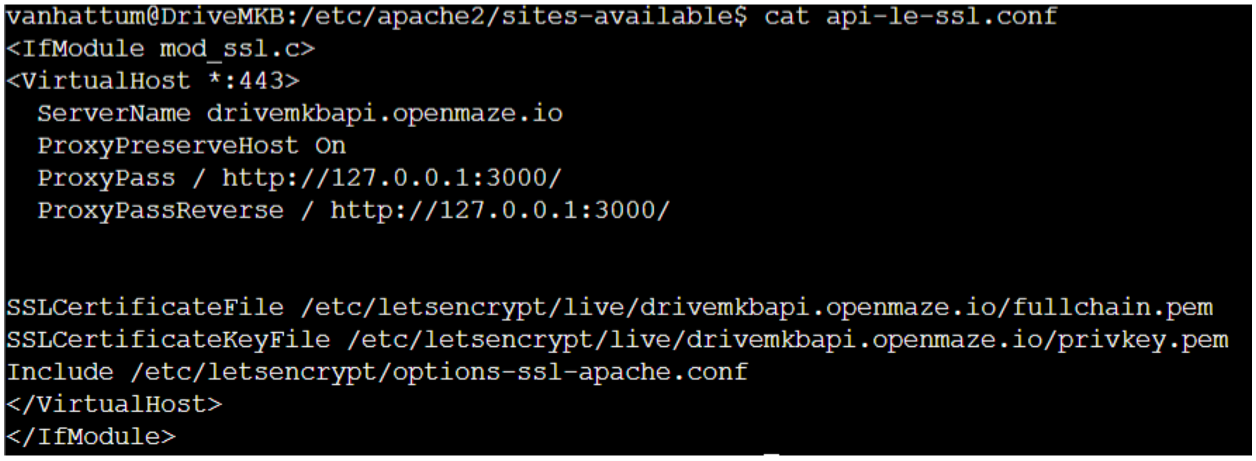
A reverse proxy must be configured on the VPS so that controlled access can be given to users from the web. Note that in this example we used two domains that must be configured to point to the VPS IP. Note also that the ports must match the ports specified in the “*.env”* and “*docker-compose.yml*“ files.



```
vanhattum@DriveMKB:/etc/apache2/sites-available$ cat frontend-le-ssl.conf
<IfModule mod_ssl.c>
<VirtualHost *:443>
  ServerName drivemkb.openmaze.io
  ProxyPreserveHost On
  ProxyPass / http://127.0.0.1:8000/
  ProxyPassReverse / http://127.0.0.1:8000/


  Header set Access-Control-Allow-Origin "*"
SSLCertificateFile /etc/letsencrypt/live/drivemkb.openmaze.io/fullchain.pem
SSLCertificateKeyFile /etc/letsencrypt/live/drivemkb.openmaze.io/privkey.pem
Include /etc/letsencrypt/options-ssl-apache.conf
</VirtualHost>
</IfModule>
```

Frontend



```
vanhattum@DriveMKB:/etc/apache2/sites-available$ cat api-le-ssl.conf
<IfModule mod_ssl.c>
<VirtualHost *:443>
  ServerName drivemkbapi.openmaze.io
  ProxyPreserveHost On
  ProxyPass / http://127.0.0.1:3000/
  ProxyPassReverse / http://127.0.0.1:3000/


SSLCertificateFile /etc/letsencrypt/live/drivemkbapi.openmaze.io/fullchain.pem
SSLCertificateKeyFile /etc/letsencrypt/live/drivemkbapi.openmaze.io/privkey.pem
Include /etc/letsencrypt/options-ssl-apache.conf
</VirtualHost>
</IfModule>
```

Backend

8. Future possibilities

As it stands the Quickscan can be used to easily give small business owners insight into how well they are doing in comparison to their peers. However, there are possible future steps that can be taken to enhance the tool even further. It can be adjusted so that it can be deployed more broadly to different target demographics, features can be added to enhance the user experience and performance, and additional admin dashboards can be made to give administrators insight into data trending. In this section these possibilities will be further explored.

8.1. Different target demographics

Currently, the Quickscan is specifically targeting the smaller entities in the catering industry. However with some effort the content, automatic data gathering and result displaying can be adjusted, and deployed separately. Currently it is relatively easy to change the questions in the survey part, with accessory advice. It takes slightly more effort to change the textual content, and finally it takes the most effort to change the automatic data gathering process.

Further work can be done to simplify the process of 'rebranding' the tool. For example a custom Control Management System (CMS) can be built in, so that it will be easier for regular administrators to change content. Moreover, the data gathering process can be refactored to be more modular and part of the CMS. This will make it easier for clients to set up and customize their version of the Quickscan, and deploy it as a separate version.

8.2. Enhanced user experience

There are some features that could be added or enhanced to deliver an even better user experience. At the moment the user is asked for their Instagram name, however it is also possible to implement Facebook Login, simplifying the process and also improving the rate limits as mentioned above.

Moreover, the user can be asked to give their email address, linking their results to their email. This makes it easier for them to be able to return to their results, while making it harder for other parties to see other result pages.

On the result page, improvements and additions are possible. Such as allowing users to filter with whom they are being compared (e.g. on region, or amount of followers). It is also possible to create a module, which shows the user how their own data is changing over periods of time, a trending graph per result.

8.3. Admin Dashboard

Currently there is no possibility for administrators or other interested parties to access all the data that is being gathered and saved. In the database all the above mentioned data is being saved, in addition other publicly available Google data such as addresses, contact information and reviews are saved. This is all saved in such a way that with every new data retrieval, the previous information is still kept with versioning. This would allow for data analysis over regions over time.

It would be a huge improvement if an admin dashboard would be developed, so that this data can be made accessible in an easy, but controlled, way.

9. Conclusion

In conclusion, this project transition report has provided a comprehensive overview of OpenMaze's journey in developing the Quickscan tool for the Centre of Expertise Interaction Design at Fontys as part of the DriveMKB project. The primary objective of this initiative was to empower small business owners by offering them a seamless and insightful means of assessing their digitalization levels.

Throughout this document, we have delved into crucial aspects of the project, including its core features, the intricate technical details encompassing the technologies employed, the deployment procedures, and the options available for customizing the tool's content. Our exploration has also extended to the identification of potential future directions for enhancing the Quickscan, thereby ensuring its continued relevance and efficacy.

In line with the overarching aim of the project, we have strived to create a resource that not only encapsulates our development efforts but also equips other parties or individuals with the knowledge and tools necessary to deploy and customize the Quickscan according to their unique requirements. This project, with its future-oriented perspective, underscores the commitment to fostering digitalization within the small business sector.

As we conclude this transition report, we look forward to seeing the Quickscan tool serve as a catalyst for digital transformation in various small businesses and industries. By sharing our journey and insights, we hope to inspire others to embark on similar endeavors, driving innovation and progress in the realm of digitalization.

10. References

11. Annex

MongoDB. (n.d.). *MongoDB: the Developer Data platform*.
https://www.mongodb.com/

NestJS - A progressive Node.js framework. (n.d.). NestJS - a Progressive Node.js Framework. https://nestjs.com/

Next.js by Vercel - The React Framework. (n.d.). https://nextjs.org/

Docker: Accelerated Container Application Development. (2023). Docker. https://docker.com/