Haaga-Helia
University of Applied Sciences

Microsoft PowerApps as an Alternative Solution for Business Application Development

Troy Palmer

Bachelor's Thesis
Degree Programme in
Business Information Technology
2020

Abstract

Haaga-Helia

2020

Author(s)
Troy Palmer

Degree programme
Business Information Technology

Report/thesis title
Microsoft PowerApps as an Alternative Solution for Business Application Development

Number of pages and appendix pages
34

The objective of this thesis is to analyse the benefits, constraints, and usability of Microsoft PowerApps as a No-Code implemented solution for a large business. When businesses require a solution to a problem through an application or service. The business would generally solicit a contracting firm for applicable services. No-Code programming has allowed for non-coding proficient business professionals the opportunity to implement company-wide application solutions. Microsoft is a leading competitor in enterprise technology, providing several possibilities for cross integration on their platforms. Although this technology is readily available to all members of a parent organization, the challenges of using this technology effectively are a factor in determining long term viability of a technical solution.

This thesis is divided into three parts: Theoretical Background, Company A Case study, and discussion of Microsoft PowerApps as a No-Code platform. This thesis will examine one case project done by company “A”, a large multinational organization. The author will conduct an analysis of project outcomes and methodologies to determine if PowerApps is a conclusive alternative to outsourcing business solutions. Usability testing of PowerApps software will be done via questionnaires distributed to Company A employees with a background in using PowerApps for development purposes. Questionnaires will evaluate experience levels of employees, accessibility to learn, familiarity with PowerApps, fulfilled project goals, completed desirable outcomes of projects, and desired features and functions of PowerApps.

This study will give an understanding of PowerApps’ current iteration in a large business environment. Ease of integration, tools available for integration, and supporting software will also be listed and explained. The results of this thesis will also give the reader a concise objective resource to understand the benefits and constraints of Microsoft PowerApps.

Keywords
No-Code Development Platform, PowerApps, Case Study

Table of contents

1 Introduction

In a traditional business environment, large companies utilize enterprise level software to manage their company's infrastructure and staff. Computer applications have been a staple in the advancement of business solutions to save money and time on tasks valued towards the consumer. This business software comes at price. Outside contractors and companies who develop solutions for large businesses, usually tailor their products for a linear set of tasks. If a client company requires a custom solution to a problem, those solutions would take time and money to develop. In a traditional enterprise level environment, clients of software applications have modest control over the scalability and adaptability of requested(software) products and services. The inflexibility can cause a shift in purchasing power by large companies who need more value for their purchases.

No-Code/Low-Code Development Platforms can be the next evolution of the application production process. No-Code software has the ability to put business development directly in the hands of an everyday user. Producing applications for businesses by businesses and directly implemented by employees. In a traditional enterprise level environment, clients of software applications have modest control over the scalability and adaptability of request-ed(software) products and services. The inflexibility can cause a shift in purchasing power by large companies who need more value for their purchases. With a No-Code solution in place employees will have the ability to adjust their productivity without needing a technical background (Waszkowski, 2019)

No-Code applications are software services that allow individuals and companies the ability to create services and solutions for their companies that require no programing knowledge. These application services are built with ease of use in mind. Application building process begins with designing your service through tutorial questionnaires to narrow down the services the application can provide. The process continues with designing through drag and drop options for: text buttons, data displays and pages. The end of the building process carries on through employee testing, data source connections (if required) and if the application can handle the task it is set out to accomplish. No-Code solutions are not just for users that have no coding experience. The goal of the desired application comes down to the application and the purpose. Such No-Code solutions could build rapid prototypes and cut down on work time for professional software developers. When software developers prototype certain systems, No-Code applications can provide a proof of concept.

PowerApps, is Microsoft's solution to No-Code development. PowerApps is the successor of Microsoft Access. Access is a web-app and database information management tool. PowerApps builds off of Microsoft's recent software as a service platform, Office 365. Access was limited to its own database system, while PowerApps can be used in various third-party platforms. PowerApps gives flexibility by giving access to several data sources. These data sources allow for PowerApps to be integrated with existing enterprise architecture without extensive knowledge on in place systems, due to PowerApps having a No-Code development environment. PowerApps is going to be the focus of the research done in this paper.

This project originated from an internship with a testing a solution with PowerApps to digitalize a manual paperwork process. PowerApps was chosen because it was already a part of Company A's Microsoft Enterprise License. This is rather contradictory to the statement of third-party developers and services not being required. But all No-Code solutions come from somewhere. Microsoft was just one of many large companies that saw shortcoming in the services they provided and adapted.

This company project resulted in prototype testing in a large corporate environment with no on-site Information Technology (IT) support. The process of this project from" Company A", has been documented and will be presented as a case study in this thesis. The purpose of this thesis is to analyze the research and data from Company A to see if PowerApps is a suitable alternative for internal application development use. As well as to see what development challenges No-Code development solutions pose, as opposed to third party suppliers

1.1 Thesis Objectives and Deliverables

The Objective of this thesis is to analyze the benefits, constraints, and usability of Microsoft PowerApps (MSPA) as a No-Code implemented solution for a large business. Businesses usually solicit contracting firms to implement software changes if the client company does not have its own support staff. This study will examine one large company working on a way to digitize its paper workflow. This will include a case study from Company A and Research from independent studies done by other researchers.

Deliverable of this project will be project integration and development results as well as user studies performed via questionnaire form(s). Using these results the author can form a conclusion regarding the practicability of MSPA as a contender for software contracting companies. Learning objectives in this study are to explore the capabilities of Microsoft PowerApps, and to compare its deficiencies in a rational manner against in-place traditional solutions.

1.2 Research Questions

A study on Microsoft PowerApps (MSPA); as an effective alternative for large scale business application development, with a single company focus.

Question(s) this thesis should answer:

1. Is PowerApps a viable solution for a large company's software challenges?
2. What are the challenges of implementing a No-Code solution in corporate infrastructure?
3. What are the expected benefits of implementing a No-Code solution for a company?

These questions will be answered throughout the case project study and review of cited literature sources.

1.3 Research Methods

This thesis focuses on Qualitative research methods for information gathering. Qualitative research is the use of data that is non-numerical by nature and done by firsthand observations and existing data (Silverman, 2015. 4-6). This thesis will use the following forms of qualitative research to answer the research questions listed in Chapter 1.2 Research Questions:

Participant Observation and Feedback

Interviews and Questionnaires

Existing Theoretical Data and Journal entries

Participant Observation was critical in the Case Study for Company A, the benefits and challenges of PowerApps would need to be experienced through the eyes of Employees using the application to express their ideas and observations. This was set up through a User Prototype Testing setup. This prototype test allowed for the employees to use the finished prototype to gage responses and reactions of how they approved or disapproved of PowerApps in their work environment. Observations were collected primarily through verbal feedback and suggestions of the participants. There was a total of nine participants in the testing of Company A's prototype. These Users were instructed to use the Application that was created to achieve the goal of submitting a digital form to the data collection system. Users were tested one at a time on a laptop to see if they could follow the visual cues needed to complete the task without aid, in a timely manner. Observations of users also allowed for employee developers to gage reactions and understand how the testing process works. Feedback was organized through open ended questions. These questions were:

How easy was it to understand the goal of the test?

How difficult was it to achieve the goal?

What would you improve in the visual design of the Prototype?

What additions would you make to the prototype that would benefit you?

How likely would you use this application in an everyday working environment?

User Feedback and responses to questions above are located under Chapter 3.5 User Feedback and Chapter 4.2 Company A in House User Study. Observations to daily work life were also being surveyed. The researcher needed to understand the normal process for the shipping requirements in the Company A case study; Referenced in Chapter 3.1 Introduction.

Initial interviews and questionnaires were given to the employee developers and employee users of the prototype application before the project plan was in place. An employee developer is an employee in a position to create PowerApps for their employees or peers to utilize. The researcher has put himself in this employee developer category as he was under contract by Company A. These questions were to gage the level of expertise for the developers and the technology experience of the users. The questions for the developers were as follows:

How long have you been using Microsoft PowerApps?

What is your programming background if any?

Was PowerApps easy or difficult to learn?

Have Past Applications impacted your company's performance? If so, how has your company's performance changed with PowerApps?

What services have you created with PowerApps? How did you use this product?

Was your project team or companywide?

Were you able to accomplish your goals with PowerApps?

What are aspects of PowerApps that you would like to change?

What is the future of No-Code/Low-Code Applications?

Is this product a cost-effective solution?

The questions above are reviewed in Chapter 4.1 in House Developer Study. User questions focused on basic questions such as experience with technology and experiences with past company PowerApps projects, found in Chapter 4.2 in House User Study.

Theoretical data is used for comparing observations, technical data and cost effectiveness of Microsoft PowerApps. Theoretical data will state the purpose of technologies and explain the functions of PowerApps as a No-Code platform. These are used as a reference for maintaining the purpose of this thesis to answer the research questions where observations and interviews from the case study will not. The researchers personal experience from the case study will be compared and contrast with other researcher/individuals who have used PowerApps in a similar study.

2 Theoretical Background

The theoretical components of this research paper incorporate several technologies and studies to form a cohesive structure in which to convey the data to the reader. This section will give highlights to concepts that will have interrelation to each other. Technologies and basic concepts are included, as well as universal and practices that are common amongst the software industry.

2.1 Rise of No-Code/Low-Code Development Platforms

The beginning of early modern no/low-code development platforms can be traced back to early days of web-page content management systems. Content Management Systems (CMS) were ways for non-tech or code savvy individuals to update content on their websites or blogs. It required no knowledge of HTML (Hypertext Mark-up Language) or CSS (Cascading Style Sheets). But before content management systems, there were early forms Rapid Application Development tools (RAD). These tools included Microsoft Access and Microsoft Excel. They are described as being early forms of RAD because of the amount of educational background needed in these early applications to garner success in your development (Rouse, 2020).

With the need for business problems to be solved quickly and there being a strong demand for skilled programming talent, No-Code based business have started to climb rapidly. This gives the providers of the technology the ability to charge for products that users create themselves usually on a subscription basis. These No-Code/Low-Code Development Platforms (N/LCDP) have evolved to the point where they can be utilized cross department for multiple tasks. Early RAD products of the past were confined to a small group. With the Aid of a Graphic User Interface (GUI) building an application with N/LCDP’s is simple. Usually utilizing a drag and drop or point and click method for building the desired application. With more advanced user able to use function commands to create more in-depth resources for their company.

2.2 Microsoft PowerApps

PowerApps is a large collection of services used to build custom applications for users and businesses. It uses a function based No-Code environment that works seamlessly with most Microsoft Office's services. These applications that are built through PowerApps are runnable on mobile devices, supporting IOS, android, and windows. PowerApps is also available on web browsers for company intranet. These applications are hosted on Office 365 and utilize integrated data collections.

PowerApps has two different approaches to creating applications. The first is a canvas driven application. The canvas application is a blank slate, allowing users to design and connect data to organize from scratch. The canvas allows you to drag and drop objects for your interface. The model driven application is data that is taken from supported data connectors. The model driven approach is used to display data that needs to be represented and organized in a presentable manner. Canvas models are used to usually perform more complex tasks. These tasks include adding data to an organization database. Model drive applications use existing data.

PowerApps can be easily integrated into any Microsoft enterprise level environment. PowerApps comes with Office 365 and associated Microsoft services. PowerApps can use and share data between any existing Microsoft product. SharePoint, PowerBI, and excel are common data storage solutions for PowerApps. With PowerApps container Application, you can seamlessly use your app over a variety of mediums.

The primary coding needed to use PowerApps efficiently is minimal and is comparable to using Microsoft excel or access. The workflow and progress made on applications is increased by being more proficient in the program (PowerApps) itself. PowerApps uses functions and formulas to complete back-end and front-end tasks.

2.3 Challenges of Implementing NCDP Applications

NCDP applications have a unique set of challenges as is befitting a new generation of technology. The familiarity of software development is probably the main challenge for new users of NCDP products. Users working in an enterprise environment who are not developers or technically inclined may find organizing and planning their application more difficult than someone with the expected experience (Shukla, 2020). Other applicable job positions just as project manager or a team coordinator would be adequate in overcoming this obstacle. Another challenge is the scaling of such an application. For instance, Microsoft PowerApps suffers from data collection limitations that are briefed more in this study. With application scaling and experience being common challenges, next comes the ability to be fluent with data sources. Data sources are any area of connection with a NCDP, in the sense of data storage or collection. Data source familiarity is usually outside the domain of the employee user of NCDP technology. Learning these technologies would require individual learning time. Employer visibility is also a concern and a challenge with NCDP's (Shukla, 2020). With the technology easily accessible company wide. It would be hard to keep track of everyone's projects and how or who would be implementing them. A solution for this would to be to limit the ability to access the application by express permission of the employer. And to keep track of project integration by having a group entity with NCDP experience managing the potential project list.

PowerApps has its own set of limitations that while are not breaking, are limiting in the current environment it was intended for. PowerApps must be run through its own application. This application in its current state is buggy and sometimes fails to launch over a wireless network. You also cannot create a tool used for external use outside of your company. PowerApps is limited to a company on site network, and the ability to publish outside of your work environment (Partially to the application launcher). PowerApps has a data management issue in its current iteration. Now, PowerApps only can search/add/retrieve 500 items from a given data source. This creates a problem for those who have larger scale projects that require thousands of items in inventory or customer records. Certain data sources require additional licensing fees. This might turn off potential buyers.

Security is also an issue that is concerning IT departments and businesses. The lack of transparency on the data being generated from applications could rise to be an issue (Korolov, 2019). If data from NCDP sources are being collected from employees working on projects, what kind of commercial oversight will they have to contend with? If any? Vender system auditing requires that the commercial product be well maintained and complies. But if a user makes an application with a security vulnerability. The demand of

the vender to be able to fix the issue is compounded by the fact that different licences/version could be used by different client companies. The last security issue is the issue with human error (Korolov, 2019). NCDP, as mentioned in the previous paragraphs do not have a certain set of standard practises which dictate the rules that employees must pertain. This leaves another human and testing auditing practise avoided by those who are not familiar with project management in the Information and Technology sectors (Korolov, 2019).

2.4 Benefits of No-Code Development Platforms (PowerApps)

The key benefit of PowerApps is that it is integrated into one of the most popular business office suites in the world. Microsoft has a large application library of integrated Applications that communicate with each other out of the box. This feature allows you to use Microsoft's other business applications in tandem with PowerApps. For example, if your company uses SharePoint for sharing data in the office. You can set your Power application to pull the information from the SharePoint server. You even can save and pull data from Excel Spreadsheets. The possibilities of data integration are numerous and are limited on the imagination of the employee in question. With Office integration Microsoft security is also built in with enterprise user accounts (Warghade 2020).

Advanced users are not shunned from using PowerApps. If you are a coding veteran there are programs that can help integrate programming functions into power apps. One of these helper programs is called flow. Flow allows you to use a set of commands to form triggers to direct commands into new actions or commands. For example. if you wished to turn a page of your PowerApps into a formatted HTML document for printing. It would only require a flow diagram to point your page to a HTML formatting template that you created or acquired.

Rapid prototyping for more advanced concepts is possible with PowerApps. PowerApps GUI is simple and you can create menu's for testing that serve more than idle screens. Even if you move out of the prototyping phase with PowerApps you can save time and money with an experienced user and avoid additional software licensing fees and contractors. PowerApps can almost connect to any type of data source to receives and present data. Some of that data is held behind a paywall. For instance, custom data sources and databases such Oracle, and Could services require an additional investment (Warghade 2020).

2.5 Costs of Application Development

Application development costs by using more traditional means can cost a company a large financial sum of investment. This can cost the company depending on their needs can be twenty-eight thousand for a single application or over one-hundred thousand for a single company developer a year (Dogtiev, 2020). This is not only a cost for the application itself, but also the equipment needed for the application to operate. In the case study for a company to operate at the standards they desire to become digitalized this would require digital scanners and specialized equipment not designed for the devices desired for the company. The equipment desired by the company is smartphones and tablets. The cost of warehouse management equipment, software and training comes to a substantial amount if built by the company itself and would cost a large amount if brought in by a contracting firm (Dogtiev,2020).

PowerApps costs can be measured by goal. For the case study prototype the goal is to have an easy to use cost effective system. The First cost of PowerApps is the price for which it is set. Currently PowerApps is free with a Microsoft Enterprise license. The company in the case study already pays for this license and utilizes most of its services. PowerApps costs money for addition features. These features include expanded support for 3rd party storage software such as Oracle database and 3rd party software language support. The cost for upgraded enterprise user rights for PowerApps development features comes to 15 euros per a month per a user (Microsoft, 2020). This fee can become somewhat high depending on how many users are running the application as well. PowerApps will only let users run the application if each Microsoft account is associated with the same license (Microsoft, 2020).

PowerApps is utilized through a Microsoft application called PowerApps available on the Android and Apple application store. This application has no cost and runs on most tablets and smartphones. If the company in question distributes these devices, then the cost is negligible. In the case study, Company A already provides each employee with a phone and tablet able to run PowerApps. To run the PowerApps application the user needs to input his work credentials associated with his Microsoft account to get started. Once the User has logged into their account, they can run whatever application they have access to (Microsoft, 2019).

2.6 Other Tools

Microsoft PowerApps allows for integration of several parent technologies. The tools/programs discussed in these sections are the ones that were used during the case study process.

2.6.1 Office 365

Office 365 is a cloud-based subscription service that supplies customers with a variety of personal and business applications. The most common Office 365 Services is the Microsoft office suite, which includes Word, PowerPoint, Excel, and Outlook. Office 365 has a user-based ID system that is integrated in all of its products through accounts. These User accounts can give accountability and tracking for companies to see what tasks on PowerApps have been completed (Microsoft 2019).

2.6.2 SharePoint

SharePoint is now a Microsoft Office integrated document management and storage system that operates from a web browser. SharePoint is highly configurable collaborative environment, with the ability to suit a variety of company needs. SharePoint has several features that PowerApps can take advantage of. One of those advantages is the ability to store information that can be easily share between employees. SharePoint Lists can create columns and rows that can store incoming information as well as be able to retrieve that same information to Microsoft products. PowerApps can target those list names and input and retrieve the information from SharePoint (Microsoft 2019).

2.6.3 Microsoft Flow

Microsoft Flow (MS Flow) is a software that allows for users to automate tasks across multiples services and applications. MS Flow allows for integration of tasks through a series of triggers. These triggers use actions committed in certain application to trigger events throughout your infrastructure. These triggers are used as feature replacements for products that cannot natively perform those actions. An example of this is that if you use PowerApps to take a photograph, PowerApps cannot natively be given instructions to duplicate itself or save itself in certain locations. Another example is that is taken out of the Case study further in this thesis. PowerApps cannot by design print itself in a presentable format. Therefore, Flow is required to trigger a series of events to create an html form to organize the data on PowerApps to be printable. Microsoft Flow has recently been changed to Power Automate (Microsoft 2019).

2.6.4 Power BI

Microsoft Power BI is a Business Intelligence service that makes use of several Microsoft products and plugins to visualize data and create reports (Microsoft 2019).. The use of Power BI in PowerApps is used for measuring Metrics of sent data from PowerApps to Company A's analytics team. Microsoft was not directly worked with by the researcher in the case study for Company A. But is worth mentioning for the context of the stakeholder.

2.7 Usability and Design

To successfully adapt and navigate a Power Application. The user will need to have the ability to easily use the application without hindrance daily. The application will need to follow usability and design rules which will enable the test users for success in navigating and controlling their environment. The application will need the following elements (USGSA, 2013):

Unity: The page must appear to fit together and be coherent
Space: To have the perceived notion of readability and improve focus
Hierarchy: Shows the significance of elements
Balance: Equivalent distribution in colour, size and placement on page.
Contrast: Emphasis on standing out, such as colours or icons.
Scale: Size range and interest.
Dominance: Focal point.
Similarity: Continuity

The elements listed above will be taken into consideration for the overall appearance of the application. The application will retain a form of similarity to the paper document it will be replacing.

3 Case Study Company A

3.1 Introduction

Company A needs a logistical upgrade to digitize the manual paperwork process for incoming warehouse products directly from factory. These incoming products are then documented and set aside for outgoing shipments to various locations around the world. The thesis author was approached by Company A to implement a solution that required no additional outside-third-party support to integrate into Company A's working environment. Company A's solution is to incorporate technology from Microsoft office 365 application suite. The proposed solution that the thesis author was asked to use is Microsoft PowerApps.

The process that Company A uses for incoming shipments from the factory is a manual filling out of receiving paperwork (See Fig.1). The information filled out are the products in the received shipment as well as several administrative items that will be discussed further in this study. After the paperwork is filled out and received it is sent to another area of the receiving facility. This requires the receiver to walk 200 meters to a local computer to manually scan and input the shipment information. This process requires two objectives to be complete with the receiving paperwork. The first objective is to take the information on the receiving paperwork and put it into various data visualization software for several separate departments in Company A. The second objective is to scan the document and save it on a SharePoint workspace. The information for the first objective must be manually typed into the target data software. The second objective is for record keeping and can be scanned at the public company computer desk for the local department. The solution that was proposed was a testing prototype that we could use for user evaluations before widespread company testing

[redacted]

Asiakirjatyyppi
Toimintaohje

Sivu
1 (2)

Otsikko		Tunniste	Versio	Pvm
Tuloilmoitus		[redacted]	[redacted]	[redacted]
Omistaja	Ylläpitää	Hyväksynyt		
[redacted]	[redacted]	[redacted]		

TULOILMOITUS

		Tuloilmoitus liitetään rahtikirjaan			Päivämäärä			
	määrä	Tarkistetaan seuraavat asiat ja merkkaa raksi, jos puutteita			Auto / kontti nro, rek.			
Lavatyyppi	↓	Kuljettajalla EI ole huomioliiviä/-takkia			PO/STO			
EUR-Lava		Kuljettajalla EI ole turvakenkiä						
CHEP-Lava		Kiiloja EI ole asetettu paikalleen						
Kerta-Lava		Luomuvaatimukset ID: 23571						
Muu-Lava								
Euro-Pan		Muuta huomioitavaa						
Preformihäkki								
CP			Valitse oikea yksikkö ja kirjaa täysien lavojen määrä					
REXAM/BALL								
FIN-Lava								

Tuote nro	Tuotteen nimi	Parasta ennen/tuotanto päivämäärä	Kilot / Kappaleet / Myksit	Irtoyksiköt	Määrä yhteensä	Täydet lavat / Vajaat lavat	Labra näyte otettu

Huomautettavaa !		
Kuorman purkaja:	Varastoiva operaattori:	Purku alkanut:
		Purku loppunut:

Figure 1: Tuloimoitus (Incoming Delivery Form)

3.2 Requirements

The challenges for undertaking this task were to create an application that accomplished both objectives mentioned and do the tasks more efficiently and electronically. Microsoft PowerApps is executed on mobile devices. Company A already provides tablets and phones to all employees, this enables all workers to have access to the application. Using a tablet connected to the company network allowed for the process of taking information without having to use a computer on the other side of the warehouse.

Storage requirements require that the data be easily accessible by the analytics team to use for power BI. The solution that was decided for the prototype was a SharePoint list to host the data in appropriate columns. SharePoint lists allow for a lot of data to be accessed by multiple Office 365 plugins and applications. This seemed appropriate for unforeseen use case scenarios with the datasets from the received products and items. Two SharePoint lists were to be created, one for the datasets and another for images of digital paperwork that needs to be filed for record.

Product database integration so the application can access the products without full user input. The user of the application must be able to search for a product and have information autofill in the form without effort and in a timely manner. Employees of company A are timed on their activity output, so speed was of a concern when filling the application form out.

The Products on the list needed to be able to have enough fields to fill in the variety of shipment items. For examples: there needs to be a certain number of rows and columns to fill different product information. The other items on the paperwork needed to include Date, Time, safety requirement for vehicle carrying shipment, vehicle number, name of receiving employee, and damaged goods. A time was also needed to be implemented to track user performance.

Photos needed to be taken of damage products. This requires the camera feature in Microsoft PowerApps as well as access to a tablet or phone with a camera, which the company is supplying. The employees must be able to comfortably and hastily be able to perform their duties and actions while using this application.

3.3 Building the Application

The first phase of implementing this solution for Company A after compiling requirements, was planning. The author to create a schedule that was within an acceptable timeframe to Company A. This schedule was instrumental to meeting deadlines and gaining access to individuals in the company that could assist in providing resources. A schedule planned over the course of six months was planned out for Company A.

The author needed to research the capabilities of Microsoft PowerApps to determine if the Microsoft service could provide the requirements for Company A's upgrade. After establishing that a working prototype was possible. Potential users of the PowerApps Applications were interviewed in preliminary rounds to give suggestions to the author about what to improve in the creation of the form on the application. This fillable form would be used on a daily basis by employees of Company A and needed to fulfill requirement pertaining to the ease of access for old and new employees.

The author began working on a general layout for the application. The application would be a single page with all the fillable fields displayed after opening the application. The designing of the physical appearance of the application was simple, PowerApps has a drag and drop ability that allows you to easily insert shapes and different types of fields. The goal of the front-end design of the application was to show a familiarity to the user. The application should look and feel much like the physical form that a user would usually submit by hand. The application form though should also be improved upon in areas that are redundant or do not make sense to the user. Once a form design was settled upon by a contingent of management and users, the functionality of the application process would begin.

A key requirement of the application was that the information entered the fillable form on the mobile devices needed to be stored accurately. SharePoint was chosen as the common storage space for the data input registered by the application. Each input line that the user would use was directly connected to a column on a SharePoint list. When the user inputted the information that was required, he would patch the information with a button that was targeted at that specific list. The application also needed to have an auto-fill feature from a product dataset. This was accomplished by importing data from an Oracle warehouse database, then copying the Oracle data into an Excel Spreadsheet and creating a SharePoint list to pull data from. So far, the user now has the ability to input company form information, retrieve products, and patch those results to SharePoint as a dataset.

3.3.1 Dataset Solutions

While building the application using Microsoft PowerApps is simple; collecting, storing, and sharing data can be difficult depending on the Company. Company A chose not to upgrade their PowerApps plan. Upgrading the PowerApps plan grants developers the rights to use outside software and tools to tailor specific needs of the application depending on the current office environment. Company A uses Microsoft Azure Cloud Platform and has access to Oracle XE as a database but was against upgrading the PowerApps plan. Fortunately, PowerApps can store and retrieve data using several other methods listed at the beginning of this thesis.

The incoming dataset used for retrieving known products that were coming in from a shipment was stored on an excel file. This excel file could then be exported to a SharePoint list that key personal would have access to edit if the need would arise. The original dataset was copied from a list from the warehouse management system (oracle based).

This data that was selected is as follows:

Article Number: The serial number in association with the product.
Article Description: The name of the product and details of its contents

Now that the inventory data was readily available. The author was able to use that (inventory) data as a template in creating the application. The second set of data that the application would handle, are the outgoing datasets. This outgoing data would be items selected from the inventory of the application from incoming shipments and all items that needed to be on the final processing form. These additional requirements for the form would come from data entered directly on the application and not from the incoming dataset.

The additional items required for the final form output include:

Safety Requirements: Safety Vest, Tire blocks, Shoes
ID Number: Drivers Identification
Car Plate: The vehicles plate number
PO/STO: Additional Identifier
Organic Requirements: does the product need to be tested
Pallet Number: How many pallets will the product be on
BBD: Best Before Dates

The additional items required cont.:

Total Amount: Of product

Discrepancy: damaged?

Quality: Arrival condition

Failed Amount: If discrepancy

Photo: If goods damaged

Time Started: Time the truck began unloading

Time Stopped: The Time unloading, and processing ended

Date: The date of time zone

Active User: Who is logged in and using the app.

These data fields would be filled in on the application itself either by multiple choice selection or fillable fields in the application. The time was recorded and coded to patch automatically at the time the form is patched to a separate SharePoint list.

This final SharePoint list from both the incoming data and the outgoing/ fillable data would be patched from the application as record for any party authorized to use the data for analytical or business purposes. Because of The Microsoft Office suites Integration, this is a seamless process for all of Microsoft's services.

3.3.2 Preparing Microsoft PowerApps for Data Integration

Once your system for data integration is prepared and the data is ready for use, the PowerApps Developer is ready to start connecting the data sources to the application. PowerApps allows you to use a blank canvas or a template. The blank canvas allows you to customize the applications physical appearance and design. The template option allows you to select from a series of pre completed apps that are designed for specific purposes. Because of the parameters of Company A, there are no templates that meet those specifications. A blank canvas will be the device to move forward in this project. After the selection of a blank canvas a new blank application will load ready for your datasets.

This was done through: *Connectors>SharePoint>Enter list URL from SharePoint site*.

Once the data is connected, it will populate the application with the imported data where the Developer can frame and organize them to their leisure. Only the Incoming SharePoint list is required for the data section. This is because the incoming and the outgoing data will be combined in a final list after the information required is entered.

3.3.3 UI Design

Company A's application needed to be designed in a particular manner so that employees could easily adapt to a new medium. That required the application to mirror the original form (Tuloimoitus) as closely as possible. While the application also being modern and inviting to new user. The researcher has signed a Non-Disclosure Agreement (NDA) as requested by company A to keep the visual data and appearance of PowerApps Application undisclosed.

3.3.4 Fields and Forms: Administrative

Once the UI design and layout are established for desired presentation of the application; the functions and formulas are utilized to move user inputted data, selected stored inventory data, and administrative data, to the SharePoint list. Administrative data was first added to the application. Administrative data was any data that was not product related to the incoming shipment. The administrative data is as follows:

Time Started

Time Stopped

Date

Active User

Safety Requirements:

ID Number

Car Plate Number

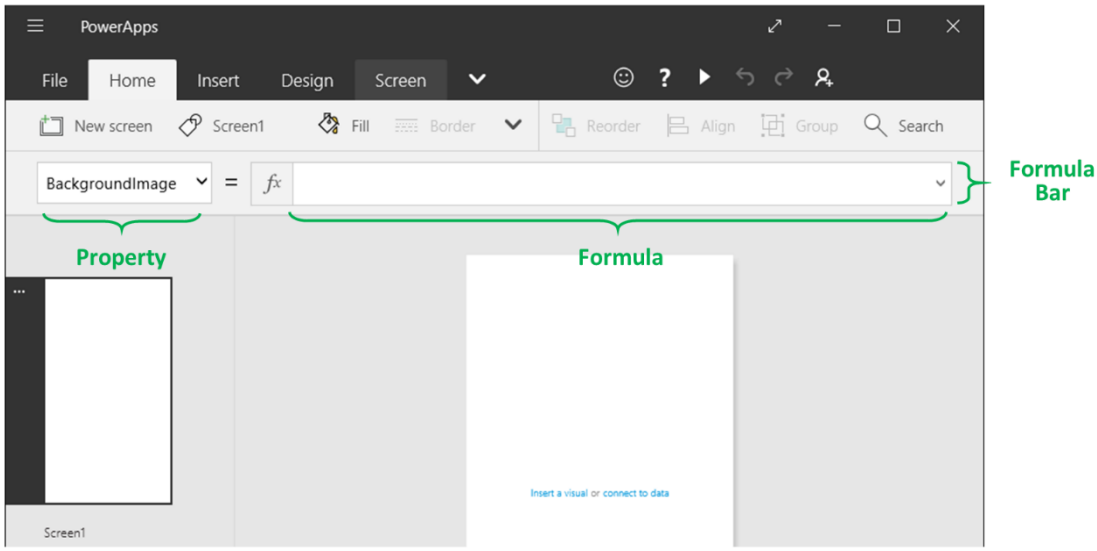

Figure 2: Formula Bar Example (Microsoft, 2019)

The time start/stop function was used to calculate how long it took each employee to process a shipment from receiving to submission. This function in PowerApps is called the stopwatch function. This stopwatch function in this application starts once the application "new form" button has been selected and ends when the data is patched with the "submit" button.

The date function was tied into the local device's date/time settings. In PowerApps this is called the IsToday function. This is not to be confused with the date control function, which is a manual setting of the date. Both the IsToday function and the stopwatch function worked together to give different administrative data. The date function filled the role of what date the shipment came in, and what day the total data was submitted.

Active User identification was used to identify the employee that was processing the incoming shipment. This user data was displayed in a text box field that was submitted with the total data. The username was imported from Microsoft SharePoint. Each user signs in with their Microsoft account when using a company device. When signed into that device, PowerApps registers your account automatically. Displaying the Name of the User in the text box field was accomplished with the function "User(). FullName".

Safety Requirements is a workplace compliant checklist that employees processing incoming shipments needed to adhere to. This prerequisite was achieved by inserting a checkbox control. The checkbox control's output was either a "yes" or "no" . ID Number and Car Plate Number are plain text fields that are inputted by the user of the application.

3.3.5 Fields and Forms: Stored Data

Stored data is the data that is being retrieved from the product inventory list that was created in SharePoint. This data was stored so that it could be called and automatically filled in an autofill text input field. This feature was designed to take less time in filling out the form so submission times could be faster. The Stored Data that was being called to the PowerApps form is the *Article Number* and *Article Description.*

The Article Number and Description needed to be searchable in a SharePoint list to be imported to the fillable form. In a standard text box this was not possible. The only function control that had a searchable list property was the drop-down menu. Two dropdown

menus were used along with the Filter, Search, and Lookup Functions. Once these functions were implemented, each corresponding drop-down box would be able to automatically fill each other.

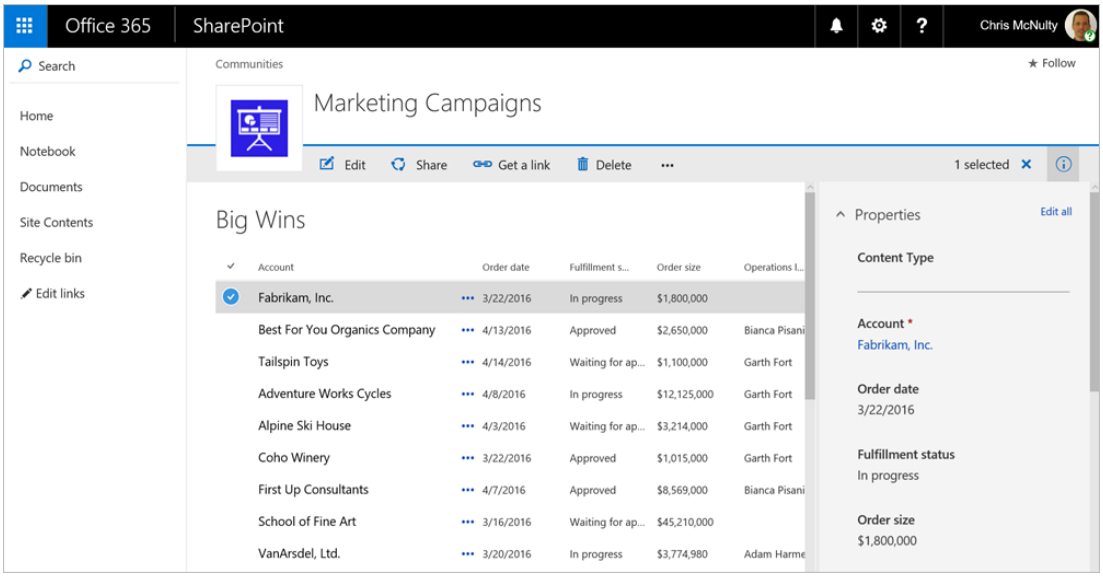

Figure 3: SharePoint List example (Microsoft, 2016)

3.3.6 Fields and Forms: User Input Data

User Input Data is everything that is filled in from the incoming shipment information. This data includes: *Total Amount, Discrepancy, Quality, Failed Amount PO/STO, Organic Requirements, Pallet Number, Best Before Dates.* These items were a combination of Textbox, checkbox, date, and dropdown functions. Each item is manually entered by the Users as they are receiving the shipment. This is done is the simplest way possible because speed is a requirement for processing.

3.3.7 Flow and Pictures

The requirement of damaged goods to be photographed and documented required a Microsoft application called Flow. As mentioned in flow's description, flow is meant to bridge gaps in Microsoft software that the original software does not process (PowerApps). Uploading pictures to SharePoint from PowerApps is currently more complicated than it should be for an easy to use program. Photographs taken by the device that the PowerApps App is saved, is stored in base64. This was a setback in development because SharePoint saves images in binary. Because PowerApps cannot convert photos on its own, flow is needed to process the workflow expression.

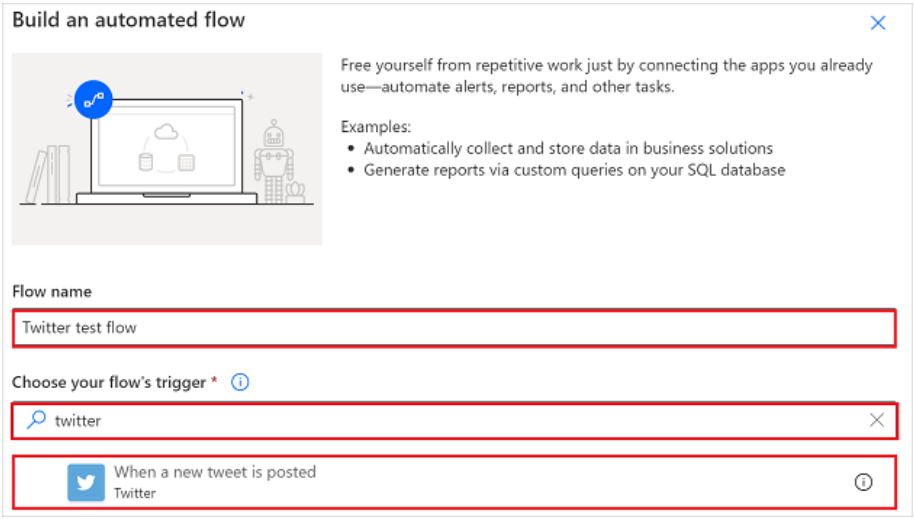

Figure 4: Flow trigger example (Microsoft, 2019)

Flow Utilizes several triggers to initiate this process. The flow trigger is prompted when the finished form button is selected. This initiates the Patching data process which will be elaborated on in the next section. If a photo is stored in the local device library that has been recently taken, the trigger will find that photograph and convert begin its process to binary. To begin you select the files and tagged functions in the application. Sending data from the app to a created file location inside of SharePoint. The first expression that will be entered into the flow process is selecting the parameters of the flow. In this case, the user wants to input an expression that will look like this: *base64ToBinary(triggerBody()['name of file content variable']).* Next you will need to target a file path, in this case the targeted SharePoint list. The process itself is not too complex but could cause issue from none code minded workers who are attempting to create picture-based applications (Microsoft 2019).

3.3.8 Patching Data

The patch function is used to amend, modify, or add new data to the SharePoint records. It is the equivalent of saving or sending data to a database. A SharePoint list is a column of items and data accumulated through user inputs. Patching data from a PowerApps requires a formula stating the coordinating name of the item to be patched and its corresponding column ID. The column ID is an identifier used to keep communing naming practices between the application and the list of data. An example of patching a record to be posted to the SharePoint ID is as follows:

Patch(Customers, First(Filter(Customers, Name = "Contoso")), { Phone: "1-212-555-1234" })

Each word before the colon highlighted, is the item name that corresponds with the list. Items in parentheses are subcategories that are targeted much like a file path on a traditional system. Patch of course is the function used to execute the command. Patching is used as a button function. This means that patching is done through a clickable button on a touchscreen mobile device through the application. More than 20 different items need to be patched at the same time. The patch command for this application is quiet long.

Editing is a task that needed to be done for redundancy. If an employee makes a mistake or forgets to add information, they will then have the opportunity to immediately fix the sent data. This is easily done by adding a button with the ability to recall all of the data. This is done without any functions by automatically recalling data from a completed SharePoint record. Once the data is recalled from the SharePoint list, they are then able to freely amend the data.

3.4 Challenges

The challenges presented in this project mainly focused on a few key areas of application development. The first challenge was finding an appropriate data storage solution. Company A did not want any additional costs to be applied to prototyping a new concept for a paperless delivery form system. The lack of funding meant that the expanded version of Microsoft PowerApps development was locked behind a paywall. This paywall prevented extensions and API (Application Programming Interface) from popular third-party data sources and resources. This required some innovated thinking in how the applications data was handled. PowerApps natively supports Microsoft services such as Excel and SharePoint. Because of SharePoint's common usage among Company A's employees, it was chosen to tackle this process.

The second challenge is pulling data from the SharePoint list in a timely and orderly manner. The search function on Microsoft PowerApps can only handle 500 items in a row at a time. The product list from incoming data holds over 2000 items. For PowerApps to be able to search through such an extensive list, each search has to be segmented in sections of 500 items. This was done through search function coding with PowerApps object function control. But this function does not immediately solve the issue. Occasionally the application would have a data set error or crash completely. Resulting in the application having to restarted through the Run PowerApps application. Sectioning the data further into smaller groups of 200 items solved the runtime and crashing issues experienced with the Application. But resulted in longer times in populating the form with the correct data. This was deemed an acceptable compromise by the staff of Company A.

The final challenge was making sure all of the data once the form was filled, was patched correctly to the SharePoint. The main problem that was a requirement given by Company A, was that the SharePoint list was not accepting photos in the same list as the rows of Data. This is where flow would come into use. Flow is used to change the file type of the picture to be compatible with viewing with the SharePoint list. This means that once a picture is taken, flow will then take the base64 data of the picture and translate it into Binary. While Flow performed its purpose, it also caused an issue while viewing the entire column of outgoing data. The issue was never solved. An available alternative was to send the translated photo data to a new SharePoint list specifically designed to view photos.

3.5 User Feedback

Employees were given an opportunity to test and give feedback on the Application during development. The first thing that was noticed by the Users was that the application did not have enough rows for incoming products. The requirement given to the developer by the company was that no more than 5 rows would be needed in an incoming shipment. What the users suggested was that they would need to be able to manually add items if there are mixed shipments. If such a case were to happen though uncommon, they would require at least 20 rows for inputting and retrieving data.

Pallet type was also a concern. Each incoming article needed a different pallet type or a choice of pallet types. Pallet types are how many items come on each individual pallet and having different types of pallets means new identifiers needed to be added. The ability to multiple pallets for each article also needed to be inputted, for a case of multiple pallets of the same type could save space and was already a feature on the paper document.

Lab results were another feature that was suggested to be added. Lab results are the testing process to make sure that certain biological contaminants in organic products were good for sale. This requires an on-site lab at Company A’s facility.

The employee users felt that certain features were unnecessary requirements. These included good/bad input option, the exception description and delivery truck carrier information. The information on the final project prototype were though still dictated by management and not the users.

3.6 Project Outcome

The project concluded with being a moderate success. The main desired features and outcomes were accomplished. The project planning process was the lengthiest and most difficult process. With PowerApps as a new software, traditional approaches to solving problems were not enough in completing the project. New methods such as agile development and comprehensive testing was established. This was a difficult aspect to press on a Company whose sector was not Information Technology related. In order for Company A to understand how to get the most out of the PowerApps service, they needed to understand how the software works and its shortcomings.

Company A wanted PowerApps to replace a massive shipping environment that has been done by hand for years. At the current state of PowerApps software state, it was not powerful enough to replace that entire system. PowerApps was able to relieve a lot of pressure on simpler tasks. As mentioned in the challenges section, PowerApps was not able to handle the large amount of data required to perform its tasks efficiently. A SharePoint List was also no suited to be a replacement for a traditional database. While a database connection with PowerApps is possible. Company A did not want to expend any resources due to budget constraints on the project.

Company A's management was also a factor in the outcome of the project. While full support was given, some features and requirement which should have been mentioned in the application did not make it to fruition. This was because of the dissonance between employees and their management. The Users of this application would be ground level employees. When interviews and testing were finally being conducted features such as: more pallet type information, more articles and items that were deemed unneeded on the form; the thesis contract was expiring. So, the new changes could not be completed. The proof of concept was established but could not be taken company wide. This is because of all requirements being a burden on the PowerApps systems, as well as not supporting ideal practices (such as database allocation and funding for additional features).

Integration with company systems was a simple process. Company A had a full Microsoft suite and Wi-Fi allocated throughout the premises. PowerApps being fully integrated into every level of Microsoft application helps its case as a No-Code solution for business environments. But business environments that do not demand as much information or requirements to be a functioning application.

4 User and Developer Studies

Users and developers were both given separate studies depending on their role in Company A. User studies focused on their experience with the application as an operator. An operator means that a user has not developed any aspect of the application but must use it in an everyday working environment. A developer in a No-Code environment is an employee who is actively trying to utilize PowerApps software to create new solutions for company productivity.

The user studies focus on the feedback and comfortability of the user of the application throughout the design and implementation process. User data was recorded by the author throughout the process to determine the challenges and benefits of implementing Microsoft PowerApps into a corporate infrastructure as a solution to software being built from the ground up for Company A.

The Developer research focuses on the feedback from the builders of the applications. Those employees who are trying to create new solutions for problems at their place of employment. These studies are focuses on the broader aspect of if the technology is ready to be integrated and accepted fully by employers as a replacement technology. This does not pertain to the case study in this paper. But in the developer's overall experience with the software (PowerApps).

4.1 Company A in House Developer Study

The developer research studies look upon the experience of individual developers to understand if Microsoft PowerApps is the right tool for the job. Three individual PowerApps developers were given a survey and then answered questions based on their experience with the software. The survey consists of general questions. Such as programming experience, product learnability, project types etc. The interview consisted of focused questions on the professional impact PowerApps had on the developer's career and workplace.

All three developers have been using Microsoft PowerApps for 1-2 years. Programming experience for 2 developers was at a no prior experience level. One developer is at an intermediate level of programming proficiency. All developers found the product (PowerApps) easy to learn. All developers were able to complete their goals with PowerApps.

When ask if Microsoft PowerApps Impacted their company's performance, all developers responded "No" and the consensus is that most of the common applications built with Company A's team were prototypes or applications with minimal function. For example: one developer created an application which stored pictures of damaged goods by taking a photograph, labelling the pictures, and storing them in a damaged folder. The damaged goods would have just been "written off" anyway and had no impact on performance. Another developer used this PowerApps to create an employee timer for work efficiency. Workers needed to record their times for certain tasks to monitor performance. This proved to be less efficient in a warehouse space where it would be difficult to acquire Wi-Fi access while in certain areas of the warehouse. The development team was asked what types of applications they have created using PowerApps. As well mentioned above, most of the items utilized by the team were for routine tasks which did not develop outside of side projects and prototypes. The only project to utilize more complex functionality would be the case study prototype.

Developers were asked if they accomplished their goals with this product (PowerApps), two out of three answered no. They answered on the basis that they had no created anything worthwhile and that the product failed them. While the third developer commented that what PowerApps accomplished was the ability to show it was possible for employees to create tools at work.

Developers were asked what they would do to improve the product. Developer 1 answered that he would improve the product by making more friendly with third party applications. As PowerApps is, it supports only a handful of outside data sources and the data sources which are supports are stuck behind paying Microsoft for upgrades. Expansion of Microsoft integration. While PowerApps is well integrated with office 365. It seems that some parts are a bit harder to connect to. For example, in the case study, PowerApps had trouble searching datasets in SharePoint, a Microsoft system.

Developer 2 said that he had no comment for improving Microsoft PowerApps. Developer 3's suggestion was that Microsoft PowerApps ownership is to strict. Ownership in PowerApps is the system in which an individual who creates the PowerApps application is the sole owner and editor. To be able to currently change ownership requires system administrator rights on a network.

4.2 Company A in House User Study

While not all PowerApps applications are built for outside use. The application built by Company A were meant to be used by all employees to ease task loads in specified areas of company business. These Users are peers of the developers and are used to experiment new ideas and solutions. The case study offered a challenge for the PowerApps developers. They had to design around workers who were used to doing their work by hand on paper and not with a tablet or phone. Design can be an objective thing. But when it comes to work, nothing beats the comfort of doing things how you would normally do them. The author had the opportunity to use the same group of nine individuals used to gather application result data from the case study; and interview them to understand the change that come with new technologies. In a traditional environment, new tools would be brought in by professional contractors, such as scanners and other specialized equipment for processing shipments. These contractors would train and instruct the employee user on how to operate the new technology. In this case, we have developers that are employees by the same company designing and implementing solutions for each other.

Employees were between the ages of 25 and 60 years old. The newer generation had no problem adapting to working by phone. The older workers preferred to work by paper and pencil. All workers are issued a cellular device such as a phone or tablet for work related business. But the Wi-Fi connection is not as fast or stable as they would like. The users feel like PowerApps is not reliable if it is not a solution implemented by software professionals. They feel like the application carries too many errors in everyday work life. These errors are blamed by users to be by their co-worker developers, which are not career developers by trade. Developer error and user error seem to contribute to a lack of trust between the two groups.

The Users or employees of Company A have discussed with the researcher that several prototypes of various project types have started to emerge from around the company. Employee developers seem to be trying to find and implement their own solutions and using their fellow peers as case studies. The developers do not seem to be acquiring feedback from the employees. The employees feel that they need to be part of the process for future application development because they will be the primary system users for future company endeavours.

Users were given the prototype to conduct mock shipment receptions without any prior background in the user interface of the prototype power application. All users were able to conduct a successful trial of inputting the required data and sending it to the correct data location. This is because the application was mirrored with minor adjustments to the original shipment form. The feedback section 3.5 User Feedback shows the initial feedback and suggestions that were placed by the employees while they were using the prototype.

5 Discussion

PowerApps has an issue with data. In the case study PowerApps main issue was that it just could not cycle through the data required to go beyond the prototype phase. The data loading issue created several other issues that took up application processing power. If one data-source could not be reached because the system could not query many results, then other data sources would also be affected. Simultaneously crippling the entire application. Counteracting the data issue was a sub searching feature that was added to the prototype. This feature was just another search bar set for smaller sets of data in alphabetical order. For example: if a user were to search a section for an item for shipment receiving, they would need to be searching 1 of several data sources connected to that one search bar. This was the only solution that possible in the time frame of the project. This solution created a longer loading time and showed to be inefficient to handwriting documents. Further developments in PowerApps processing structure will have to be achieved by Microsoft for this feature to truly show what a useful tool PowerApps could Become.

The photo mode of PowerApps seemed to be overly complicated for what it (PowerApps) needed to do. Photo's needed to be addressed separately and took a large amount of time to implement. The requirements referenced in Chapter 3.2 state that damaged shipments needed to be recorded and attached to the corresponding datasets. This was highly difficult which the programming languages not translating well and needing another Microsoft Application to solve (MS Flow). Microsoft needs to address the issue of native pictures being saved and standardized in a readable format by its own systems. This would seem to be a major oversight by Microsoft.

User Feedback and interviewing was very eye opening from a developer perspective. A large flaw in the use of PowerApps is individuals developing the application as employees. Without a development background, employee developers do not properly test or consult the userbase on how they feel or perceive the application to provide full context of the user's role as referenced in Chapter 4.2. This could be rectified by providing Usability testing education through PowerApps resource pages.

PowerApps is designed to accommodate other business departments to synchronize workflow or complete minor tasks. In PowerApps current iteration, it can create many applications meant for internal office use. PowerApps specializes in linking datasets together and displaying information. The applications for this are numerous. But as stated, is severely limiting due to data constraints.

6 Conclusion

The goal of this thesis was to answer three research questions on Microsoft PowerApps as an effective alternative for large scale business application development, with a single company focus. The outcomes of the thesis are that in application PowerApps performs its function as a small-scale substitute for conventional software development. PowerApps excels at displaying and manipulating datasets from several different data sources. But falls short on large scale projects to replace industry standards.

In PowerApps current state of development, this product is only viable on the specific needs of the company. For Company A, PowerApps is not a viable solution for the company's current software needs. As referenced in Chapters' 3.4 and 4, challenges of information loads cause a problem with querying data. PowerApps can only handle a set number of searchers for information in SharePoint. This problem is also evident in on site database management systems stated by Microsoft (Microsoft, 2019). Company A requires an information search of over four-thousand different products for shipment receiving. PowerApps is not able to create new products that are not defined in a SharePoint list or database. Unless individually inputted by the user, defeating the purpose of the ease of access requirements set by the company. In a different use case scenario PowerApps would be able to handle a small business needs if the company operated in the same manner. Large companies would not get the responsiveness or compatibility an independent software contractor could provide.

Challenges of implementing a No-Code solution in a corporate infrastructure is corporation dependent. In the case study of Company, A, the infrastructure challenges were non-existent. Company A has an ideal PowerApps environment. Company A has a Wi-Fi network that covers the corporate compound. This allows for all devices that are network compatible to be able to access the secure cloud and access the appropriate Power-Application. Company A had an appropriate level of access to Microsoft PowerApps by having an enterprise level license that supported all employees to be able to access the PowerApps application. Companies that do not have the luxury of already having the enterprise Microsoft suite would be required to change their current infrastructure to accommodate one application.

The expected benefits of implementing a No-Code solution for a company was low cost and higher productivity. Company A's digital shipping form does not meet all the requirements. Low cost requirement was met by the basic infrastructure and software licences

being in place. Low costs in manpower and training by the applications easy to use function. Low cost in equipment needed as it was already provided to the employees and developers by the company. Productivity was hindered by PowerApps conflicts with information datasets. The system could not handle such data on a such a great scale.

References

Bhangarl, S. 26 June 2020. T-Mobile Manages Company Wide Initiatives and More With Microsoft Power Platform, Available at: https://powerapps.microsoft.com/en-us/blog/tmobile/ Accessed: 11 Oct 2020

Dogtiev, A. August 28th, 2020 How Much Does App Development Cost? Available at: https://www.businessofapps.com/app-developers/research/app-development-cost/ Accessed 6 Nov 2020

KissFlow 2018. The History of Low-Code Platforms: How Development Changed Forever. Available at: https://kissflow.com/rad/low-code/history-of-low-code-development-platforms/ Accessed 10 Nov 2020

Korolov, M. June 24th, 2020. 4 security concerns for low-code and no-code development. Available at: https://www.csoonline.com/article/3404216/4-security-concerns-for-low-code-and-no-code-development.html
Accessed 12 Nov 2020

Leung, T.2017. Beginning Power Apps: The non-developers guide to building business mobile applications. (ISBN: 9781484230039)

Microsoft 2020. PowerApps and Power Automate Licencing FAQ Available at: https://docs.microsoft.com/en-us/power-platform/admin/powerapps-flow-licensing-faq
Accessed: 10 Oct 2020

Microsoft 2019. How do I Find and run apps? Available at: https://docs.microsoft.com/en-us/powerapps/user
Accessed: 10 Oct 2020

Microsoft 2019. Get started with canvas-app formulas in Power Apps. Available at: https://docs.microsoft.com/en-us/powerapps/maker/canvas-apps/working-with-formulas
Accessed: 30 July 2020.

Microsoft 2016: Modern SharePoint Lists Available At: https://www.microsoft.com/en-us/microsoft-365/blog/2016/07/25/modern-sharepoint-lists-are-here-including-integration-with-microsoft-flow-and-powerapps
Accessed: 17 Aug 2020

Mines, C. 4 November 2019. Predictions 2020: More Changes For Software Development. Available at: https://go.forrester.com/blogs/predictions-2020-software-development/Accessed: 12 Nov 2020.

Rouse, M. January 2020. low-code and no-code development platforms. Available at: https://searchsoftwarequality.techtarget.com/definition/low-code-no-code-development-platform
Accessed: 7 November 2020

Ruparelia, N. May 2010. Software Development Lifecycle Models. ACM SIGSOFT Software Engineering Notes. 35, 3, Available at: https://dl.acm.org/doi/abs/10.1145/1764810.1764814
Accessed: 5 Nov 2020

Silverman, D. 2016. Interpreting Qualitative Data, pp. 4-6.

Shukla, J. January 2020. Case study: A Government App Built with Microsoft Power Apps and Flow. Available at: https://www.gravityunion.com/blog/2020/1/powerapps-case-study
Accessed: 14 Nov 2020

U.S. General Services Administration, 2013. User Experience Basics. Available at: https://www.usability.gov/what-and-why/user-experience.html
Accessed: 14 Aug 2020

U.S. General Services Administration, 2013 Visual Design Basics. Available at: https://www.usability.gov/what-and-why/visual-design.html
Accessed 14 Aug 2020

Warghade, R. 20 May 2020. Advantages, Disadvantages And Limitations Of PowerApps. Available at: https://www.c-sharpcorner.com/blogs/advantages-disadvantages-and-limitations-of-powerapps
Accessed 16 Nov 2020

Waszkowski. R. 2019. Low-code platform for automating business processes in manufacturing. Available at: https://www.sciencedirect.com/science/article/pii/S2405896319309152
Accessed 12 Jun 2020