Getting started with Power Apps

HAMK
HÄMEEN AMMATTIKORKEAKOULU
HÄME UNIVERSITY OF APPLIED SCIENCES

University of Applied Sciences thesis

Information processing education, Hämeenlinna University Center
autumn, 2021

Petri Lehto

HAMK
HÄMEEN AMMATTIKORKEAKOULU
HÄME UNIVERSITY OF APPLIED SCIENCES

Computer training

Hämeenlinna University Center

Summary

Factor Petri Lehto

The year 2021

Job title Getting started with Power Apps

Directors Lasse Seppänen

SUMMARY

The purpose of the thesis is to find out what things a new or novice user needs
it's worth learning about Power Apps first. Using Power Apps is
becoming more common and many people are learning to use it. Learning and practicing often take time
without clear goals, instructions and guidelines. There are many instructions and
information on the Internet, but it is not always clear what the user should know and learn
first for its use.

The knowledge base of the thesis consists of own experiences, instructions collected from different sources and
of the practices I have found to be good. The thesis is functional, but also contains
investigative part.

In conclusion, it can be stated that to the Power Apps program and its functionalities
you should familiarize yourself as much as possible before starting to use it for errors
to avoid.

Keywords Application, model, connection, function, tree view, preview

Pages 41 pages and 2 pages of appendices

HAMK
HÄMEEN AMMATTIKORKEAKOULU
HÄME UNIVERSITY OF APPLIED SCIENCES

Degree Program in Business Information Technology

Hämeenlinna University Centre

Abstract

Author Petri Lehto

Year 2021

Subject Getting Started with Power Apps

Supervisors Lasse Seppänen

ABSTRACT

The purpose of the thesis is to find out what things a new or novice user should learn about Power Apps first. The use of Power Apps is becoming more common and many people are learning to use it. Learning and practicing often takes time without clear goals, guidelines, and guidance. There are a lot of instructions and information on Power Apps online, but it is not always clear what a user should know and learn first about using it.

The knowledge base of the thesis consists of the author's own experiences, instructions gathered from various sources and the proven practices found to be good. The thesis is functional, but it also includes a research part.

In conclusion, users should familiarize themselves with Power Apps and its functionalities as much as possible as possible before using it to avoid errors.

Keywords Application, model, connection, function, tree view, preview

Pages 41 pages and appendices 2 pages

Glossary

Application	A program made with Power Apps
Model	Ready-made templates for making various applications
Contact	Defining a connection to an external data source
Tree view	Presentation of the content of the application
Preview	Testing the application
Control object	Button, text field, etc.
A feature set	Control object-specific definition, for example color or size
Formulas	Among other things, they perform tasks
Flow (Automate)	Automated workflow
Procedure	A predefined subroutine
Drag-and-drop	Graphical user interface function

Content

Attachments

1 Introduction

Power Apps is an application that the user can use without deep coding or programming skills to make various custom applications for your own or your organization's needs. drag-and-the drop-style user interface makes it possible to make applications quickly. The app can be connected to several different data sources with standard connectors provided by Power Apps.

The use of visual programming applications such as Power Apps has increased is popular because the desired application can be made by a person or a group without coding knowledge and the desired application can be implemented without traditional development process with all stages.

Getting started with Power Apps is quite easy and there are plenty of instructions. However, making an application is usually started by trying different functionalities, in which case, through trial and error, you get a somewhat functional application. The application however, it is worth starting by thinking about the need and planning based on it. Different it is worth familiarizing yourself with the functionalities before using them, in order to avoid errors.

The use of Power Apps can be started by trying different applications and making use of them Power Apps ready-made templates and instructional videos found online. Without a clear goal however, adopting things can seem haphazard. After a short training session and with weak basic knowledge you may have to develop the application and functionalities to find instructions or different solutions online.

Microsoft's Power Apps web pages have a lot of information about, for example, control objects and functions, but how can this knowledge be applied in practice in making applications?

This is where the idea started to take shape, what would be the first thing to learn about Power Apps of use. What are the basic knowledge and skills that should be learned first and know? The things that I think best describe Power were selected as research questions Issues and challenges related to starting apps.

- What is Power Apps and what can you do with it?

- What should be learned first?
- How do we go about implementing the application?
- What is the effect of good planning and practice on the end result?

2 Microsoft Power Apps

Power Apps is part of Microsoft's Power Platform, which also includes Power BI and
Automated. Power Apps is an application that allows you to make custom applications that
are connected, either to the underlying information environment or to various online or local ones
to data sources. Microsoft has strongly developed the Power Platform and brought it in the past
building blocks available to application developers also for non-programming
for professionals. Fast application development and implementation without expensive and time-consuming
development projects have increased the popularity of Power Apps.

Using Power Apps does not require deep coding or programming skills, and the user can start from scratch
to make or use even complex applications for your own or the organization's needs
several ready-made models. A drag-and-drop style user interface enables applications
doing it quickly without coding. With Power Apps, you can make applications for different needs
and they can be used with a browser or mobile devices. It can be connected to one or
to several data sources and can be used to display, edit and store information. Information sources
can be for example database tables, Microsoft Excel, SharePoint lists and many others
formats. Data sources can be located locally or in the cloud, such as One Drive for Business or
DropBox.

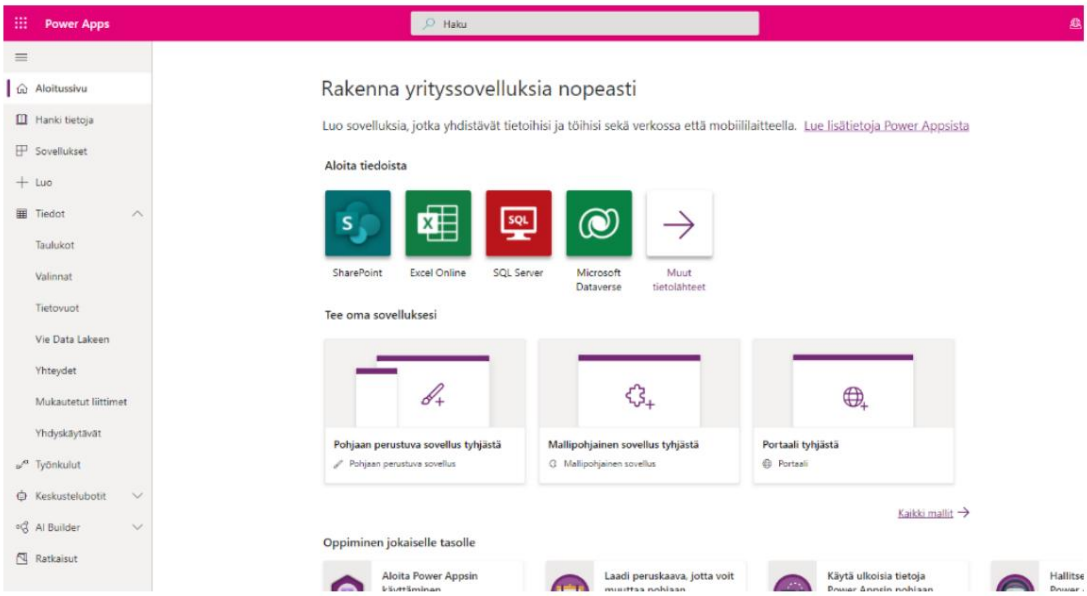
The benefits of using Power Apps include, among other things, fast implementation, streamlined development,
easier application creation, standard connectors, O365 integration, mobile readiness and
Microsoft Flow. One of the biggest benefits of Power Apps is fast application implementation. Many
the stages of normal application development, such as making the application's user interface
implemented in Power Apps with a drag-and-drop interface.
Using Power Apps streamlines making an application. The application can
to be designed, made and shared by a person or group who needs it, without tradition
the application development process with all stages.
One of the hardest parts of app development for non-professionals is knowing how to connect
from the formation to the data used. With Power Apps standard connectors, many standard
data warehouse services are easy to connect to the application. Standard connectors are available e.g.
to SharePoint, OneDrive and Outlook.

With O365 integration, utilizing data in e.g. SharePoint, Excel or other
It's easy in the O365 app. Information can be retrieved or stored and used
for example, in making reports with the Microsoft Power BI program.
The Power Apps user interface is used to create applications that work on the desktop as well as
as mobile too. In both cases, creating applications is easy and fast.

2.1 Power Apps user interface

When you start Power Apps, a page opens where you can start a new one
making an application, opening a previously made application or managing matters related to applications,
such as connections. Figure 1 shows the Power Apps screen.

Figure 1 Power Apps

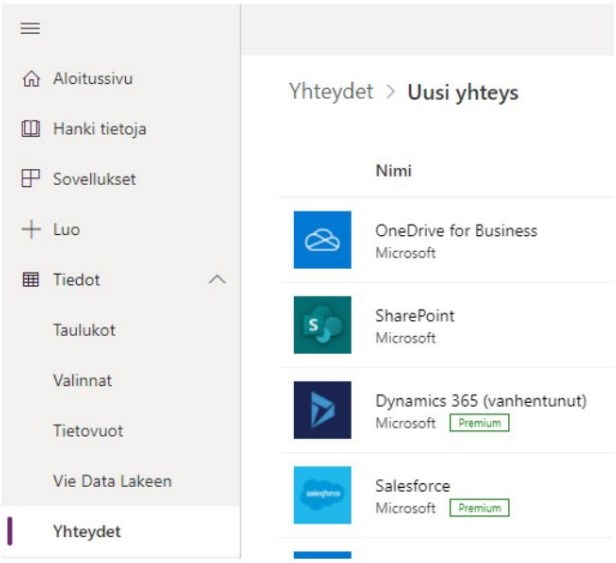


2.2 Connections

Most applications in Power Apps use external data sources that are stored
to cloud services. A common example is a table in an Excel file stored on OneDrive
for Business. Applications connect to data sources using connections and they
is opened if necessary (picture 2). Connections made to data sources can be used
for reading and writing data, such as Microsoft Excel workbooks, SharePoint lists,

SQL tables and many other formats. These can be stored in cloud services such as OneDrive for Business, DropBox and SQL Server. (Microsoft, nd-c)

Figure 2 Creating a new connection



2.3 Collection

Collection can be used to store data in the application. The data is therefore not saved to the data source, but are available locally. To create a collection is used Collect function and the collection can have multiple columns. The collection can be saved, load and clear using different function which are SaveData, LoadData and Clear Collection. The collection can be used, for example, to temporarily store the data, which you want to print and after printing the collection can be easily cleared.

2.4 Workflow

With workflows (Power Automate) you can create logic that executes one or more task when an event appears on the screen in the application. For example, can be specified button so that when the user selects it, the item is created in the SharePoint list, an email message or a meeting request is sent, the file is added to the cloud or each of these. Any control can be configured to start a workflow that continues execution even if Power Apps is closed. (Microsoft, n.d.-f)

There are various functions in the workflow, which, however, do not necessarily cover what is desired functionality, so it's worth considering these possibilities when designing the application restrictions. For example, there is no such functionality that can be used to save data. to the workstation.

2.5 Workflow and procedures

A procedure is a program module stored in a database that performs a predefined task task when it is called. When you want the application to run in a relational database a procedure that is not possible with the application's functions should be considered utilization of the procedure. In a relational database, which is managed using the SQL language, you can run pre-stored procedures. For example, a database may have a procedure which deletes field information from Power Apps based on the passed key. Power Apps has made workflow (Flow) that calls and passes key information to the procedure.

2.6 Examples of applications

The application can e.g. search, modify and enter data into the database. Apps can be e.g. online store, service desk, customer surveys or statistical processing tools. By using workflows, the application's usage possibilities can be expanded, for example to save files to SharePoint, OneDrive or send data to email.

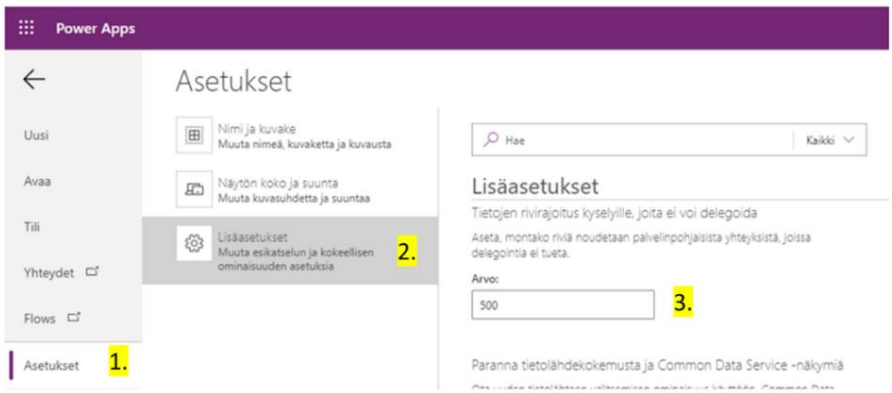
2.7 Restrictions

Power Apps may have limitations that prevent the desired application from being made. In some cases, an alternative implementation method can be found to circumvent the restrictions. Restrictions may be related to, for example, the number of records to be used, the implementation of printing or workflow possibilities.

2.7.1 Number of records

In Power Apps, the application has a limit of 500 records by default, which can be in advanced settings increase to 2000 records. This means that Power Apps is not aware of those records that exceed the value specified in the settings. As shown in Figure 3 record count setting.

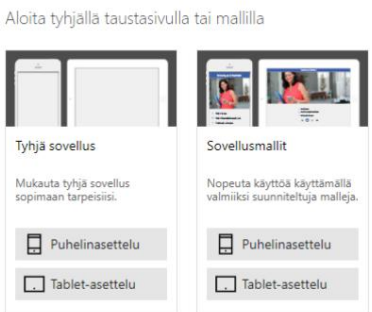
Figure 3 Image of the additional setting of the number of records



2.7.2 Phone or tablet

The application can be made suitable for either a phone or a tablet. If the application is desired works in both, we have to make two different applications. In both cases making the application takes place with the same user interface and possibly the same ones with standard connectors, but taking into account possible differences in appearance. Figure 4 shows the possibility to choose either phone or tablet layout, with which to make the application in each case let's begin.

Figure 4 Phone or tablet layout



3 Purchasing and starting to use the program

The Power Apps program can be installed, for example, from the Microsoft Store or Google Play from the store. Depending on the need, a license can be purchased that allows one user to use one application at a time (Per app plan) or a license for each user regardless of how how many applications are used (Per user plan). Power Apps license is also part of O365- to the order, so it can be used through the employer or place of study.

On July 1, 2021, Microsoft announced a change in pricing and licensing, which will take effect on July 1. October 2021. Per user plan will cost $20 instead of the previous $40 and the Per app plan will cost $5 instead of $10.

3.1 Information and instructions

There are good instructions online for getting started using Power Apps. They can be found, among other things From Microsoft websites, YouTube videos and blogs. Microsoft has a wide range documentation with a description of Power Apps, information on project planning and about making an application. (Microsoft, n.d.-g)

3.2 YouTube videos and blogs

Good situational solutions can be found in YouTube videos by the following authors, such as Shane Young *(Microsoft PowerApps - YouTube,* n.d.), April Dunnam *(April Dunnam - YouTube,* n.d.), Brian Knight/Pragmatic Works *(Learn PowerApps - YouTube,* n.d.). Also on Microsoft's website there are videos. *(Microsoft Power Apps - YouTube,* nd) Matters and tips can be found in various about blogs *(PowerApps | Microsoft Power Apps - Blog,* nd; *POWERAPPS-Aiheiset Blogit - Blogit.Fi,* n.d.)

4 Making an application

Making an application starts with its design. If we make an application for a customer or
business needs, you can find information about design on, for example, Microsoft's website.
(Microsoft, nd-h)

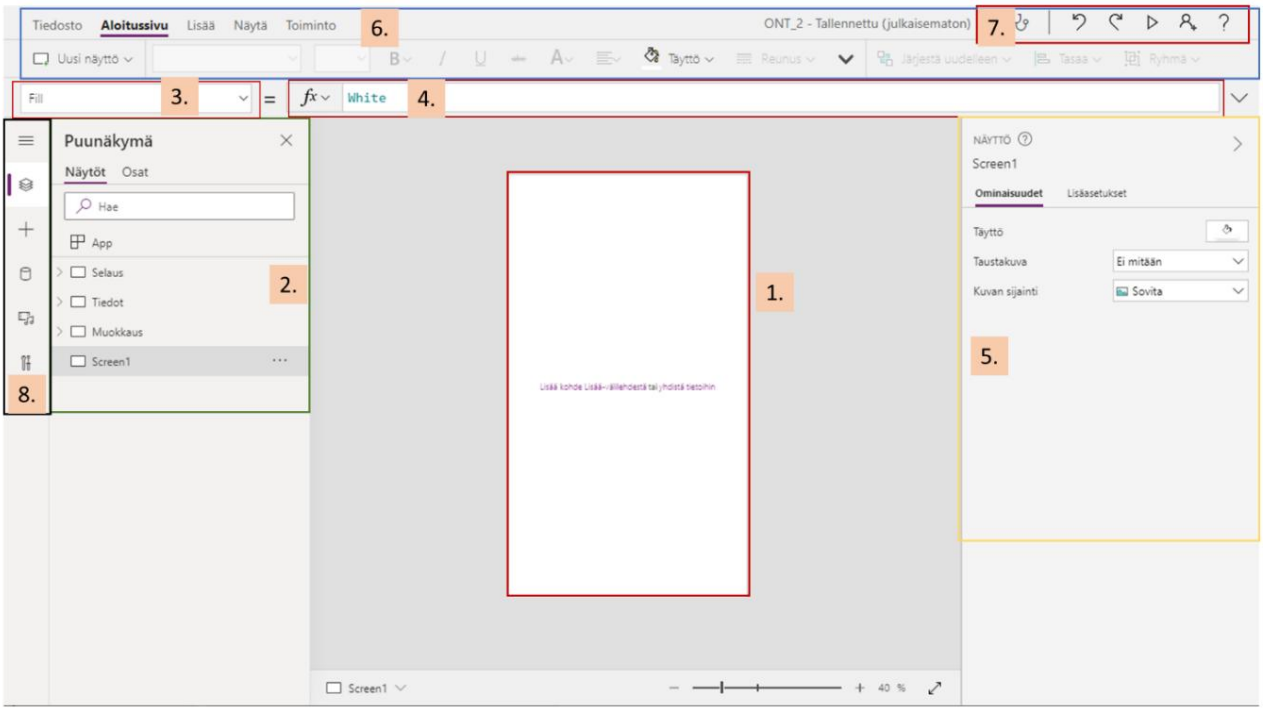
In general, before making an application, you should think about its purpose and
to plan what functionalities are needed for it. How many screens are needed, if
an application only for viewing information or also for editing? What transitions to different screens
sometimes needed and how do different objects behave in different situations? Do you want to print the information?
Is there a specific color scheme to be considered in the design?

The design can be based on the requirements, experiences and
process images. With good planning, the implementation of the application progresses logically and differently
the functions are done in a good order. In more complex ones
In applications, the larger corrections made afterwards are usually laborious because changes
may have to do to and between several control object functionalities
addictions.

5 Power Apps Studio

Power Apps Studio is used to design and make desired applications. Its most central parts are menus, screen, property pane, property list, formula bar, application functions, content creation menu and tree view (Figure 5).

Figure 5 Power Apps Studio

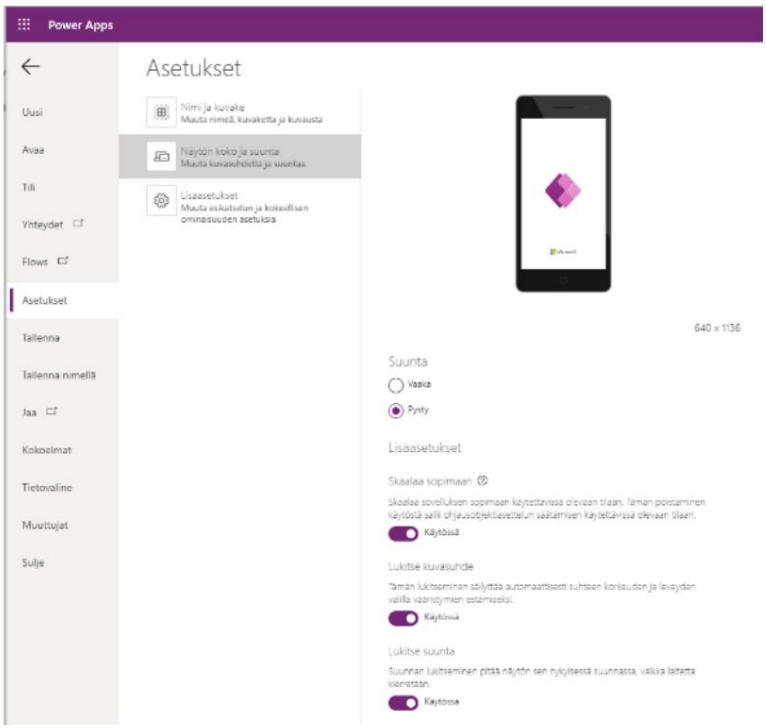


1. Display
2. Tree view
3. Feature list
4. Formula line
5. Properties box
6. Menus
7. Application functions
8. Application production menus

5.1 Settings

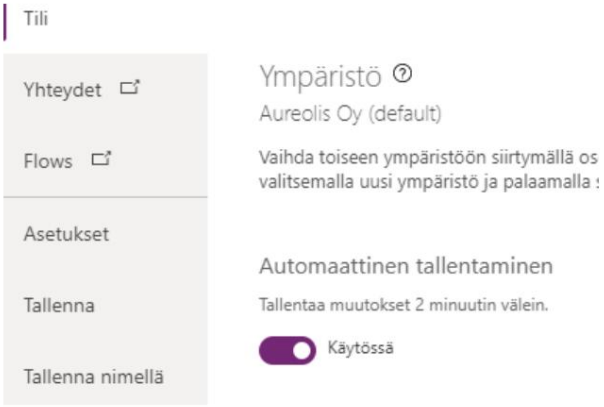
From the settings, you should pay attention to the settings related to the screen size and orientation. By default is that the screen orientation is vertical, the application scales to the available space, the aspect ratio is locked and screen orientation is locked. Figure 6 shows the Settings tab.

Figure 6 Screen size and orientation settings



The Account tab of the File menu can be used to disable automatic application saving, when you want to manually save the changes made. (Microsoft, nd-e) Figure 7 shows presented Account tab.

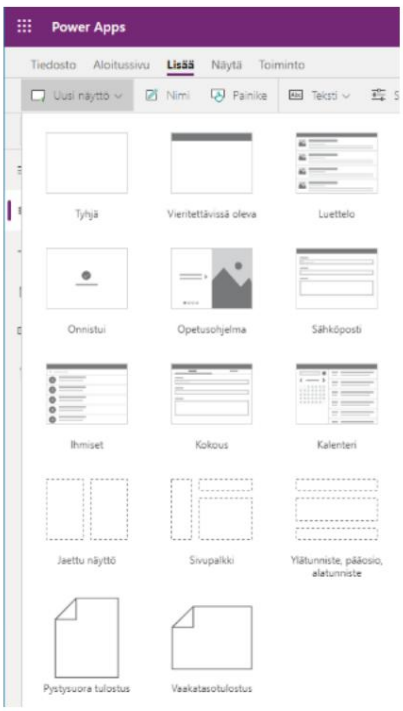
Figure 7 Automatic recording setting



5.2 Display (base)

The screen is used for creating the structure and can be added to it, for example database fields, buttons, text inputs, images, icons and charts. While doing new screen, it can be either empty or for example a predefined list. The app can contains several different screens. Figure 8 shows the options for the new displays.

Figure 8 Predefined display options



5.3 Tree view

The tree view is used to manage application elements. The tree view shows everything for the application added screens, objects, etc. When an element is selected in the tree view, it is displayed on the screen activated.

In the tree view, several objects can be made into a group and given a name. If there are a lot of objects in the tree view, similar ones can be grouped, for example objects makes it easier to use the tree view. The group can also be managed as one as a whole and hides the objects selected in it on the screen using the Visible property.

5.4 Control objects

The application consists of screens and control objects added to the screens. As control objects buttons, images, text fields, tables, etc. added to the application's screens are called.
Control objects are used to create various functions in the application and each control object appearance and use can be modified with its feature. Control objects have different types functions and some functions can only be used with certain controls.
For example, the SetFocus function, with which you want to focus the user's action on the specified one control object, can only be used with button, icon, image, label or textinput control objects.
When you select a control object, the functions available for it are displayed in the properties list. Explanations of the available controls can be found in Microsoft from the website. (Microsoft, n.d.-a) Table 1 explains some of the controls usage possibilities.

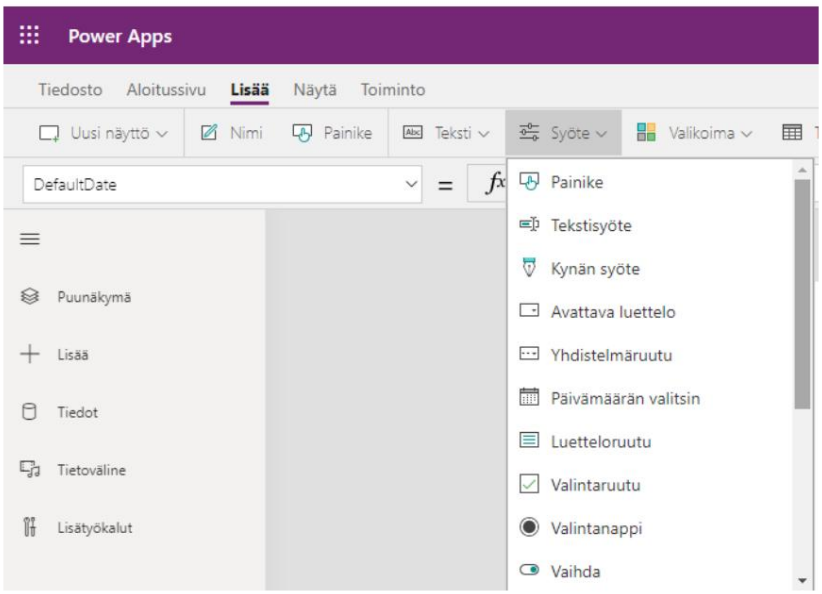
Table 1 Control objects

Handle	The button control object can be used to, for example, move to another screen or save the selected record.

Text input	Text, numbers and other information can be entered with the text input. The text input can be saved in the database.
I chose the date	The date selector is set date.
Selection box	The checkbox can be selected and its value can be set to true or false. You can, for example, select them in the selection box the records you want to collect in a collection and further print.
Timer	The timer can e.g. create a delay in the execution of a function or change the properties of an object after a certain time. For example, if you want that transition to the other page does not happen immediately, a delay can be created with the timer.

Some of the control objects are of the input type and can be used depending on the definition for example, to add data to a data source. Figure 9 shows the contents of the Input menu.

Figure 9 Input menu



Icons are control objects that the user recognizes well and can deduce which
the action takes place on them. The most commonly used icons are e.g. add, edit, search,
reload and trash. The functionality of the selected icon should be considered logical,
for example, using the edit icon for search functionality is not logical.

Various control objects are used in the application, they can be e.g. previously mentioned
buttons and text fields. In the feature list, you select which feature is selected
you want to modify or implement the object. The content of the property list is the one selected at the time
according to the control object. For example, with a button, its content is not different from
in the selection box.

5.4.1 Features

The property pane contains all properties related to the selected control. In Table 2
some features are shown. The properties of the object and their contents are also shown
in the advanced settings of the property list that appears on the right side of the screen when the object
is chosen. The desired feature can be modified in the formula line or in additional settings.

Table 2 Different properties of objects

OnSelect	The OnSelect property and the associated function define what happens when an object is selected. For example, what happens when a button is pressed.
Visible	The Visible property and the associated function determine how the object appears in different situations. For example, when a button is pressed, other fields can be hidden in terms of the logic of use.
Default	The Default property defines what the object's default information is. Definition

	can be made more complex by using functions.
HoverColor and HoverFill	With HoverColor and HoverFill properties edit which colors are used when pointing to a button with the mouse, for example.
X and Y	The X and Y properties can be used to define the position of the object on the screen

5.4.2 Tab index

To improve the user experience, it is recommended to define control objects on screens tab indexes. Indexes make use smoother when the user can tab to move to the next control object.

5.5 Functions

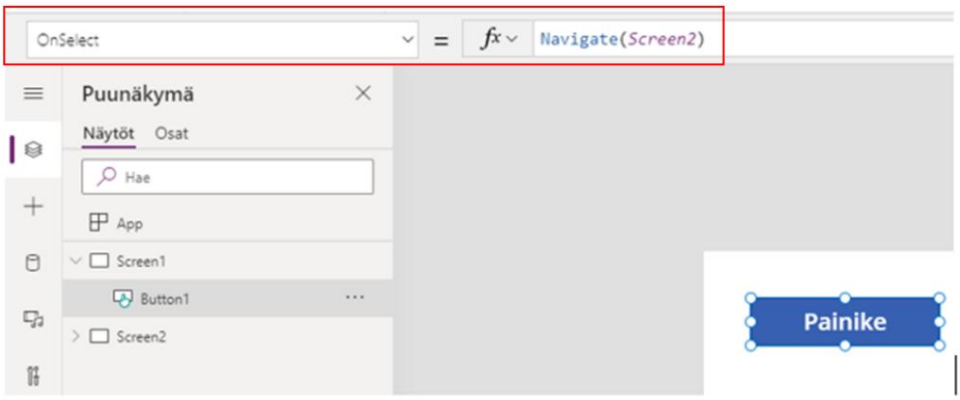
A syntax can be made in the formula line, where functions are used to define what happens, for example, when the user selects a button or other object. Functions can be used transitions between different screens, create collections, retrieve information from a data source, etc. (Microsoft, nd-d)

The use of functions essentially involves a list of features. For example, if you want to add functionality for the button, which performs the transition to another screen. In this case, we choose the functionality of the button is OnSelect and a function that implements this is added to the formula line. List of functions on Microsoft's website. (Microsoft, n.d.-b)

The formula line is IntelliSense compatible. When you write on it, you get instructions above it about the selected function and hints for writing the syntax. The formula line also gives options

of available functions. (Microsoft, nd-i) Figure 10 shows an example of a button
of the related feature and function.

Figure 10 Example of a button's feature and function



5.5.1 Navigate and Back functions

The navigate function is used to change the displayed screen. For example, by adding a button-
to the control's OnSelect functionality of this function, can be defined to another
switching to the screen. In the Syntax 1 example, we switch to the browsing screen. Syntax 2
in the example added another argument how the transition happens.

Syntax 1 Example of using the Navigate function

```
Navigate
```

Syntax 2 Example of the second argument of the Navigate function

```
Navigate(Browse; ScreenTransition.Fade)
```

When the application has several screens, you can return to the control object with the selected Back function
to the last screen opened.

5.5.2 UpdateContext function

The UpdateContext function can be used, for example, to display a control object or
to hide (Visible property) using the variable true or false in the function.
The value given in the variable would be used for the Visible property of the desired control

in the function. Syntax 3 example of using the UpdateContext function, where for the test variable is given the value true.

Syntax 3 Example of the UpdateContext function

```
UpdateContext({test:true})
```

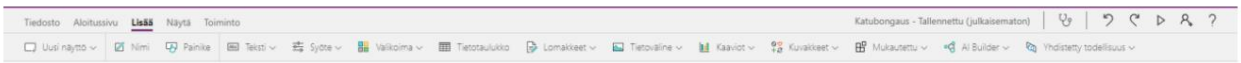
5.5.3 Collect, Clear and ClearCollect Functions

The Collect function creates a collection of records. One way of use is, for example, for a collection printing stored data. The collection contains information only as long as the application exists open. The Clear function deletes the records selected in the collection. This function only works in collections. The ClearCollect function removes the records in the collection and adds to the same new records to the collection.

5.6 The Add menu

The Add menu contains functions that can be used to add a new screen or different ones control objects. Figure 11 shows the functionalities of the Add menu.

Figure 11 Add menu functionalities



5.7 Recording and Publishing the Application

When making the application, the changes made to it are saved and the implementation takes place by publishing the application. When changing the application, the changes are not available in a published application before republishing it. When the application is released, at the same time also give people rights to it. (Microsoft, n.d.-e)

6 Testing and Review

When making an application, it is worth testing its functionality as the work progresses. The detected errors are easier to fix right away. Testing is easy to do **with the Preview application function.**

With the application's check function, you can see, among other things, possible errors in formulas.

In Figure 12, when the application check function is opened, a red dot indicates an error existence. In Figure 13, the app's check action shows that the button object there are errors in the syntax.

Figure 12 Application check function

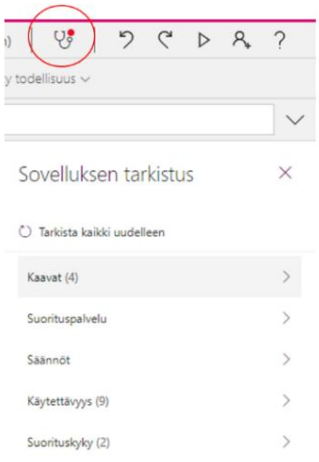
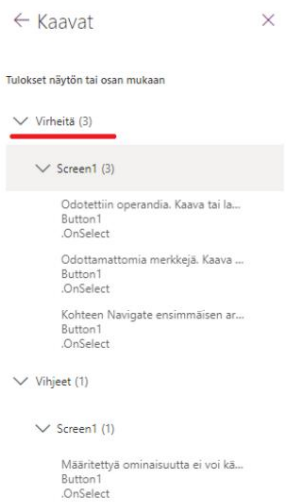


Figure 13 Errors in syntax



7 Functionality of the application

A well-executed application has the necessary functionalities, it works quickly and is logical and easy to use for the user. One of the most important issues related to usability is the functionality of the application. Listed below are things that slow down the app's functionality:

- Number of data sources
- Dependencies between screens
- Number of control objects
- Repeating the same formula in several places
- Use of unnecessary record fields
- Use of unsupported browsers
- Use of images
- Use of attachment files
- Location of files

8 Example application

The example application has used the issues presented in the theory part as well as functionalities and aims to clarify their use. The idea is to make a mobile application for street bouncers, where you can record for each street, which part of the street has been visited, when the street is bongattu, coordinate and additional information. Tampere's street names from the open source have been used as data from the data.

In Power Apps, it is important to understand browsing, information and modify the functionalities of the screens and therefore in the example application they and theirs dependencies between are done manually. Task applications are usually based on information for displaying and editing and implementing these basics is worth it understand first. When these become familiar, you can take advantage of Power Apps' automation in making screens.

8.1 Street gambling

What is street banging? Street hustlers walk the streets of the city in alphabetical order and most of the time, the background of the hobby is breaking routines and the desire to get to know the city differently way. Usually, the streets are traversed in alphabetical order and pictures are taken while walking for example, from a street sign.

8.2 Data source

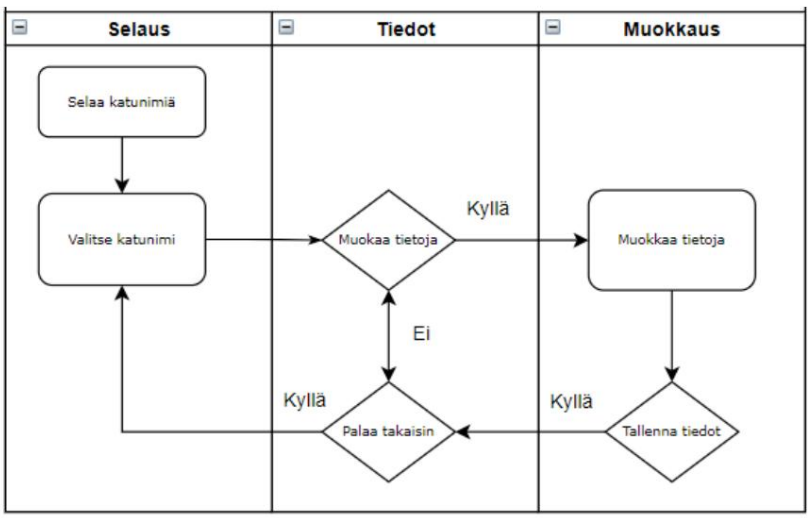
The Street Name material used in the example application is stored in an Excel workbook and it is saved to OneDrive. The street name material is connected to the sample application OneDrive- with connector. Joining OneDrive may require user login.

8.3 Application design

The user must be able to select the desired street to see all its information and to edit them. For these functionalities, three are made to the application different screen and they are browsing, information and editing screen. The browsing screen is selected

for information, street name and bonga date. The information screen contains everything related to the street name information and on the editing screen this information can be edited. Transitions between different screens and others events take place on the screens with the selected control objects (buttons). Figure 14 has a preliminary plan of the screens and functionalities needed for the application.

Figure 14 Initial functionalities of the application on different screens



8.4 Making the Application

Open Power Apps in a browser (https://powerapps.microsoft.com/en-us/) and sign in in. Let's make a new application with nothing ready. Creating a new application let's start by selecting the Base-based application from scratch (Figure 15). The application is given name and shape (Figure 16). After this, Power Apps Studio opens (Figure 17).

Figure 15 Creating a new application

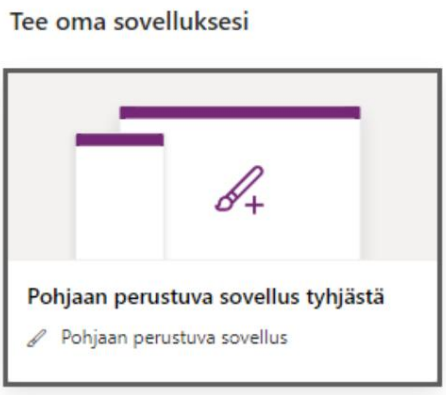


Figure 16 A base-based application from scratch

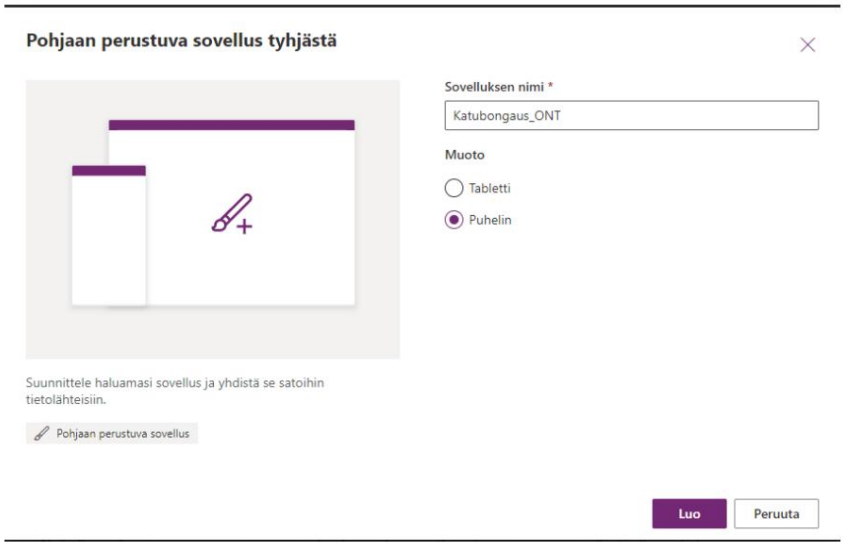
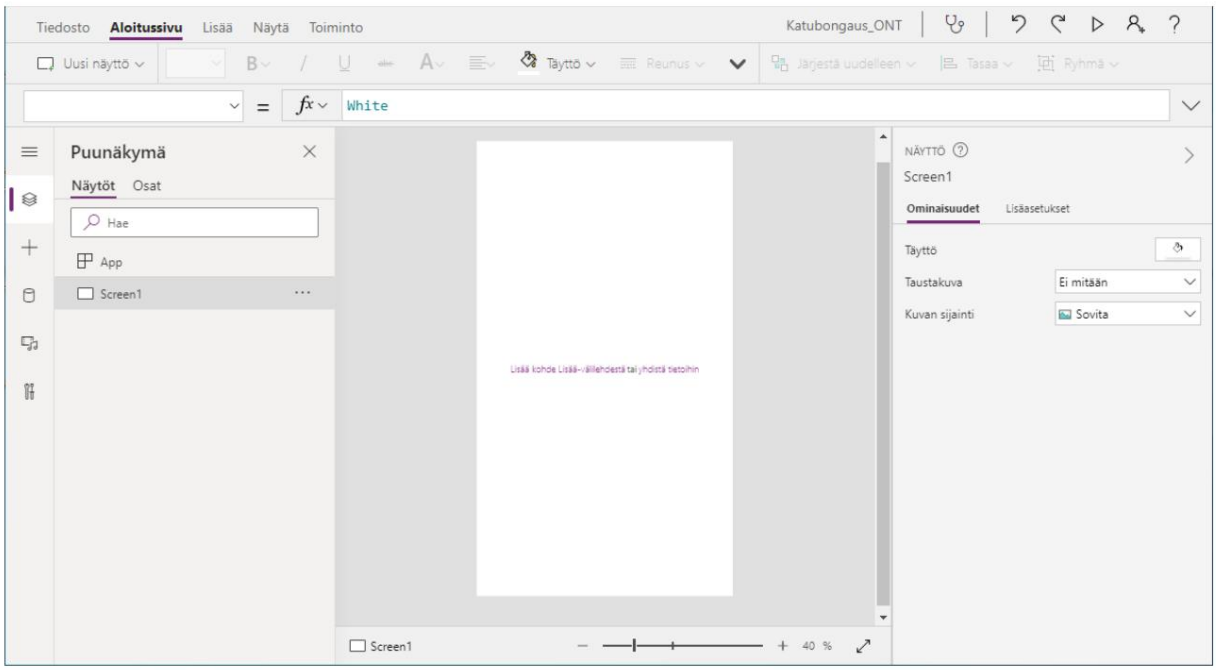


Figure 17 Power Apps Studio



8.4.1 Combining data with the application

The application currently has no data and only one system-generated display called Screen1. Connect an Excel file to the application by choosing View, Data sources and More information (Figure 18). After that, OneDrive is selected as the connector, because the Excel file is stored there (Figure 19). If necessary, log in to OneDrive.

Figure 18 Adding information to the application

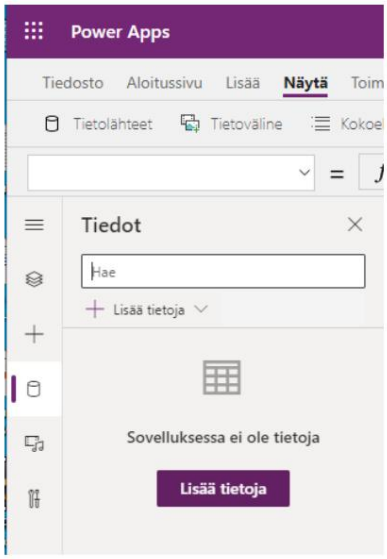
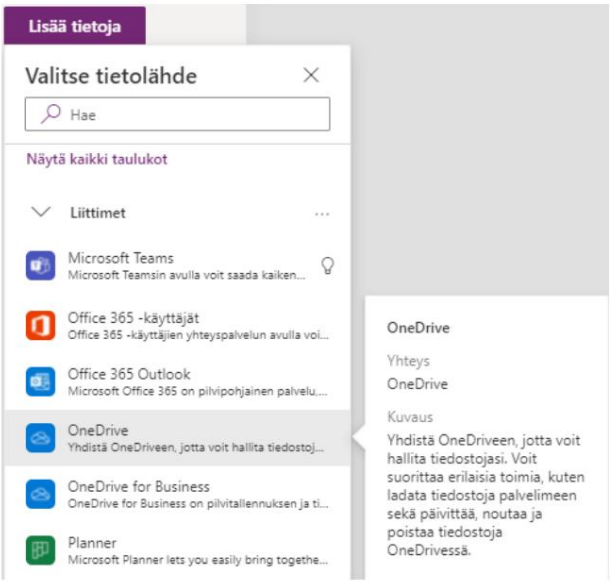


Figure 19 OneDrive connector



After that, select the desired file and the tab of the file where the material is. excel the file is called Trekadut and the file has only one tab called Table 1 (picture 20 and Figure 21).

Figure 20 Excel file

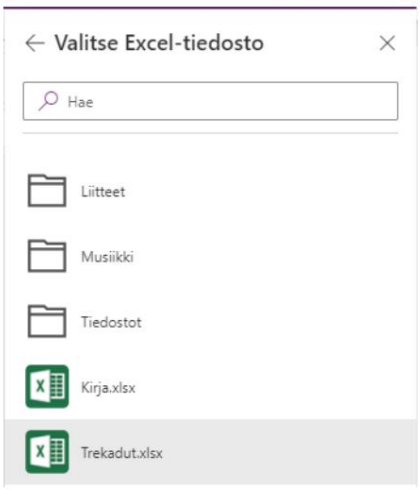


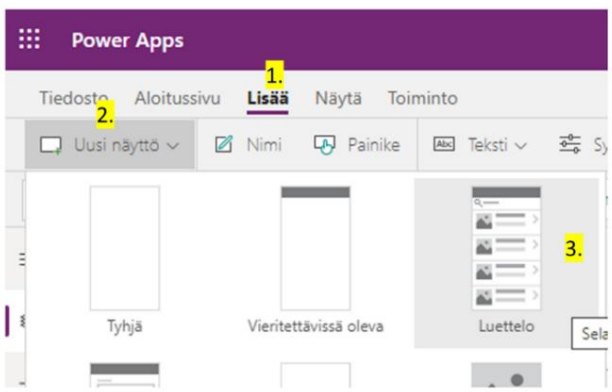
Figure 21 Excel file tab

Now an Excel file has been connected to the application, which contains the desired material.

8.4.2 Browsing screen

We want the application to be able to browse street names and add a new screen, which is of the type list (Figure 22).

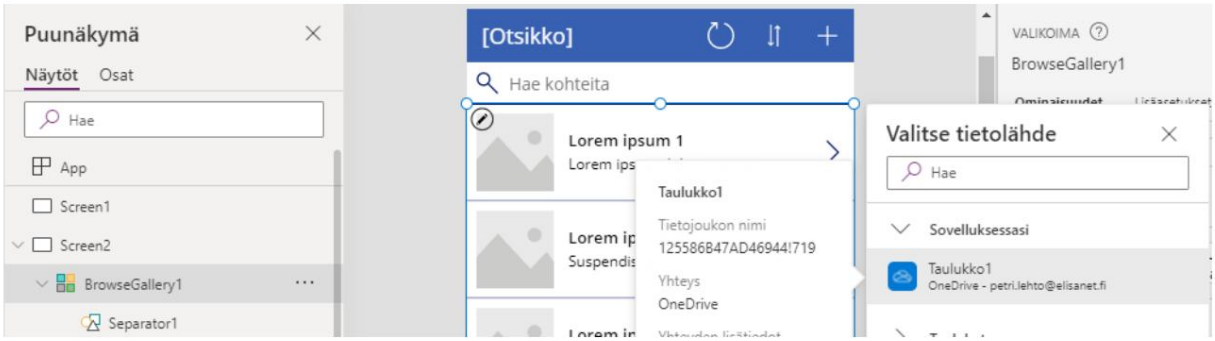
Figure 22 Adding a new screen



The new screen, of type list, already contains a place for the title, control objects and list (Figure 23). In this ready-made list, there is an option to add from each record picture and two pieces of information. The list does not show the data in the Excel file because it does not exist in it incorporated. The connection is made by activating the list, which opens the data source check box (Figure 24).

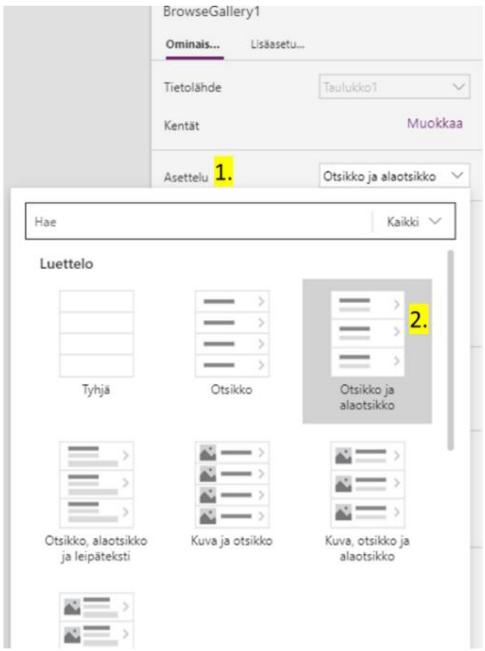
Figure 23 List

Figure 24 Connecting Excel data to a list



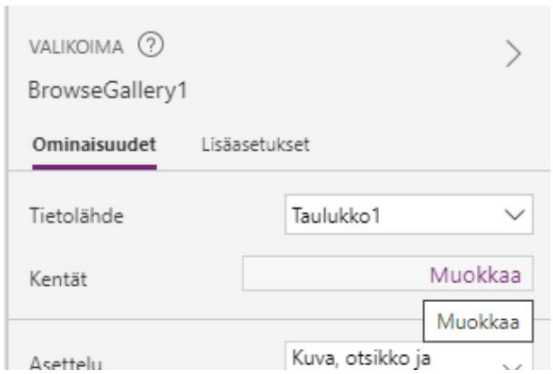
The Excel data is combined into a list and after this you can choose which data is in it displayed from the record. Since the application does not show images, a list layout can be used change to show title and subtitle. This is done in properties, Layout- from the menu (picture 25).

Figure 25 Changing the list type



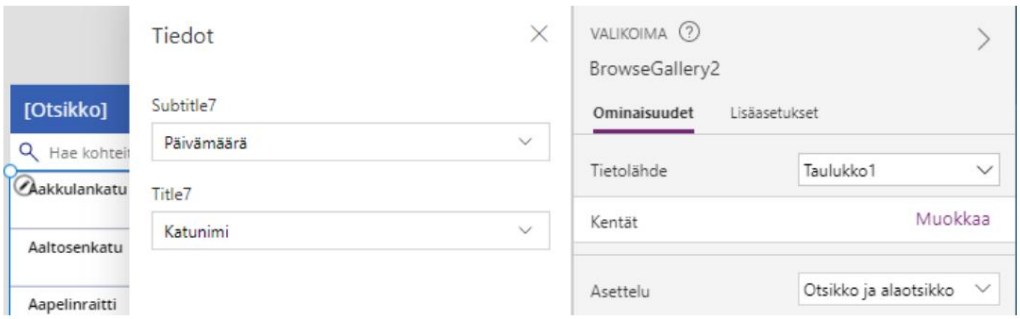
In the properties of the list, the information can be accessed from the Edit menu, where the desired ones can be selected fields of the record (picture 26 and picture 27). The properties now also show to which data source the list is merged.

Figure 26 Edit menu in properties



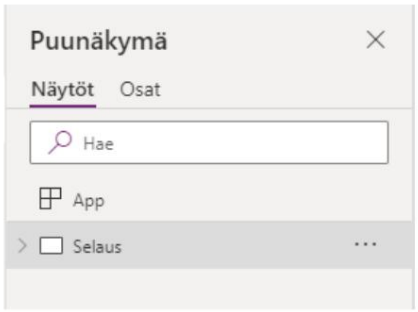
Browsing on the screen you want to see the street name and date.

Figure 27 Selecting fields



You can delete the first screen for no reason and rename the screen containing the list in the tree view as **browsing.** After this, the application has one screen called **browsing** (Figure 28). When creating the application, it is recommended to name the control object, screens, etc. descriptively. as this makes it easier to use them in the formula bar. The program helps with function syntax in writing by giving hints when you type the text.

Figure 28 Screens in the tree view



8.4.3 Number of Records

The application now has a limit of 500 records by default, let's change this to 2000 records, so that all the records in the Excel file are available. The change takes effect when the application closed and reopened.

8.4.4 Search field

Searching for information on the browsing screen is done in the search field, and the search is only targeted to the street name. The search field is made to work when the Items property of the list write the syntax using the Search function (syntax 4).

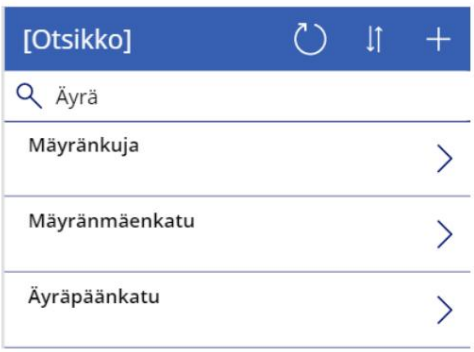
Syntax 4 Data retrieval from the Last name field

```
Search([@Table1]; TextSearchBox1.Text; "Street Name")
```

The search function is used to search from the Table1 data source, from the Last name field, to the search field (TextSearchBox1) typed information. The search targets the data source because the data is not in the application and the search result is displayed in the list.

Let's test the functionality of the search using the preview function. Figure 29 shows a part browsing screen and the search field in it.

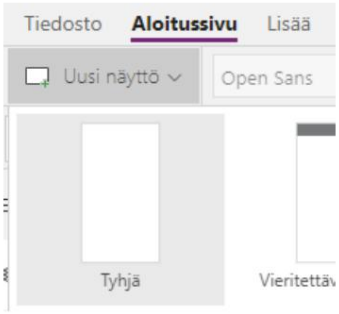
Figure 29 Testing the functionality of the search



8.4.5 Details screen

In the application, we want to view all the saved street names of the street name selected in the browsing tab information and for this a new screen is added, which is of type show. Because the street name does not have a lot of information doesn't need a scrollable screen, so a blank screen is enough (picture 30).

Figure 30 New blank screen



Rename the new screen in the tree view as **information.** Let's add a form to the screen (picture 31) and connect it to the data source. All data source fields are added to the form (Figure 32).

Figure 31 Form addition

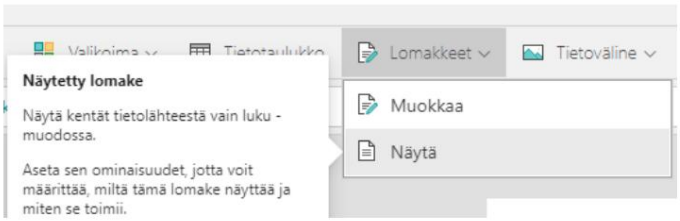
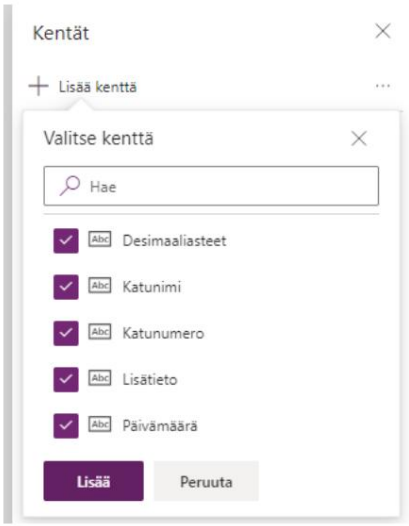


Figure 32 Data source fields for the form

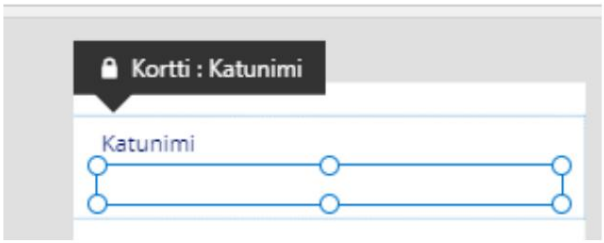


You can change the order of the fields in the Fields menu by dragging them to a different position.

Let's enlarge the cards on the screen (DataCard) so that the title and the actual field are visible

(Figure 33).

Figure 33 Card with form



Information on the form on the screen does not show the information of the selected street, because two things are missing.

The form needs to be told which street name information needs to be displayed and kept on the browsing screen

tell when to go to the information screen.

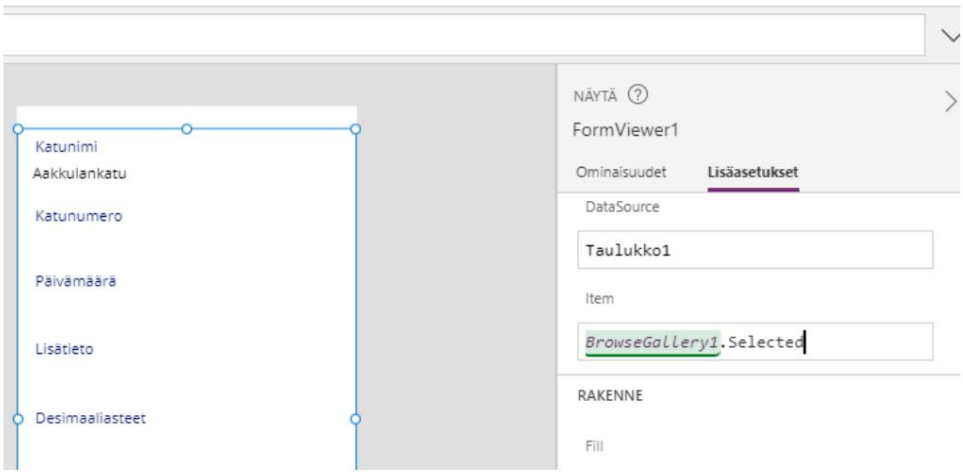
Let's add syntax (Syntax 5) to the item information in the advanced settings on the form on the information screen (fig

34).

Syntax 5 Displaying information about the selected street name in the information form

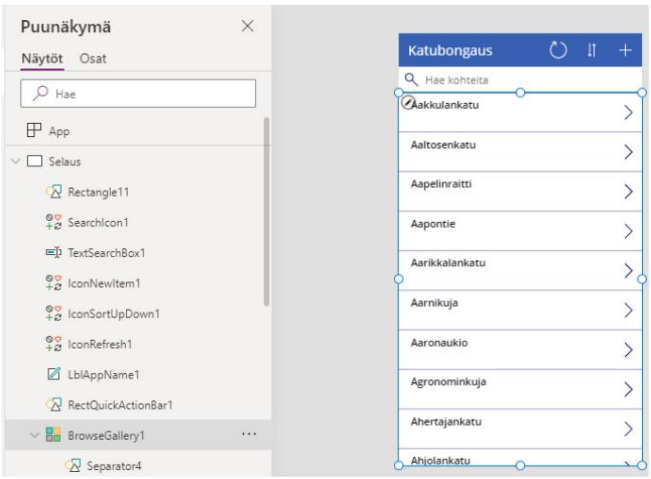
BrowseGallery1.Selected

Figure 34 Item



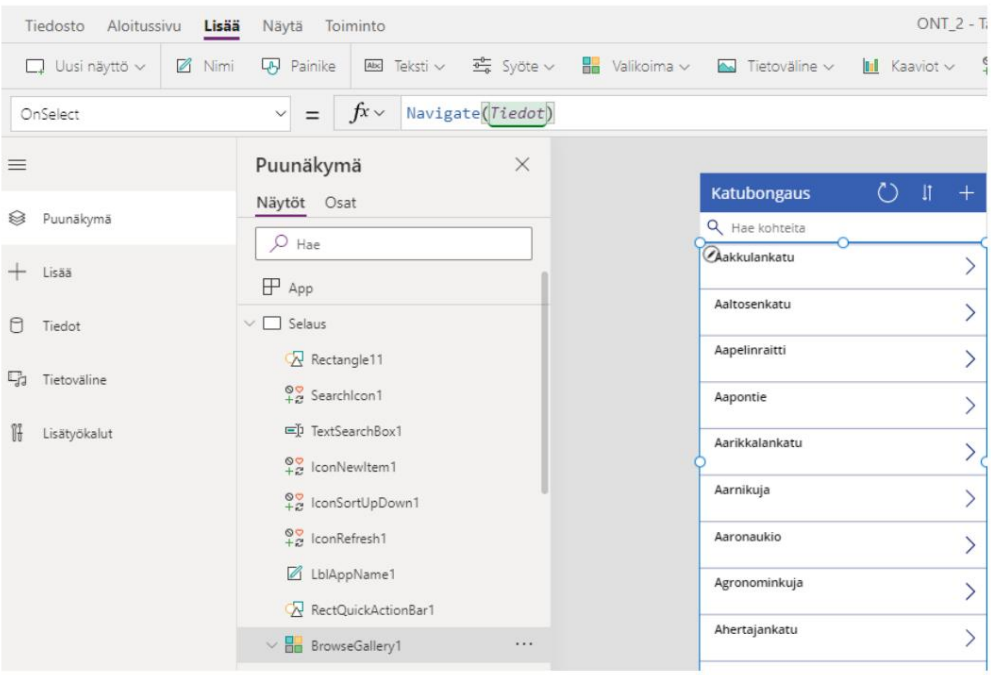
BrowseGallery1 is the name of the list on the browse screen in the tree view (Figure 35). Now that the street name is selected on the browsing screen, the information screen shows the street name information.

Figure 35 BrowseGallery1



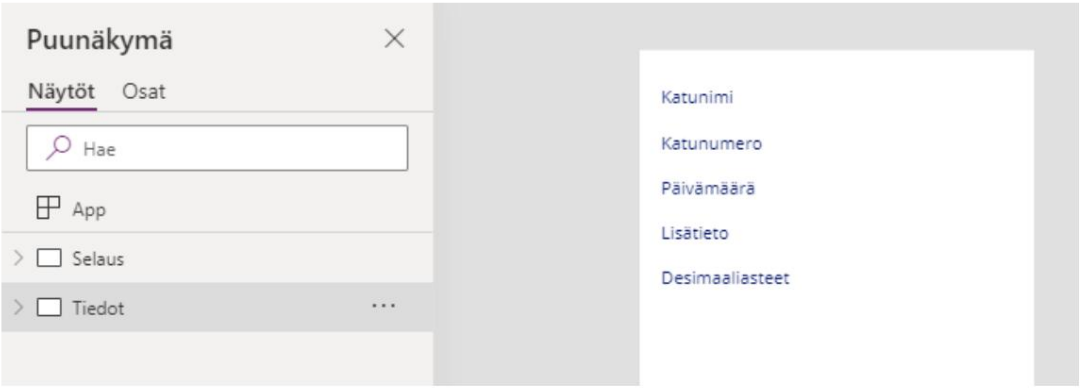
Add to the browsing screen BrowseGallery1, OnSelect property Navigate(Information) (Fig 36). Now, when a street name is selected from the list on the browsing screen, the transition to the information screen takes place.

Figure 36 OnSelect



The application now has two screens called **browsing** and **information** (Figure 37).

Figure 37 Browsing and information screens in the application



On the screen, we want to be able to move to edit the information or go back to browsing to the screen. Special buttons are made for these functions. Let's transfer the information on the screen fields down to position the buttons at the top of the screen. The fields are located on the form, so it is not possible to transfer an individual field, but the entire transfer takes place by moving the form on the screen (picture 38).

Figure 38 Moving the form on the screen

Let's add two buttons at the top of the screen and name them Back and Edit. A button the text can be changed by double-tapping it or in the Text field in the properties (Figure 39).

Figure 39 Adding a button

The back button is used to switch to the browsing screen, and for this a button is added

Syntax for the OnSelect property using the Navigate function (Syntax 6).

Syntax 6 Go to the browsing screen

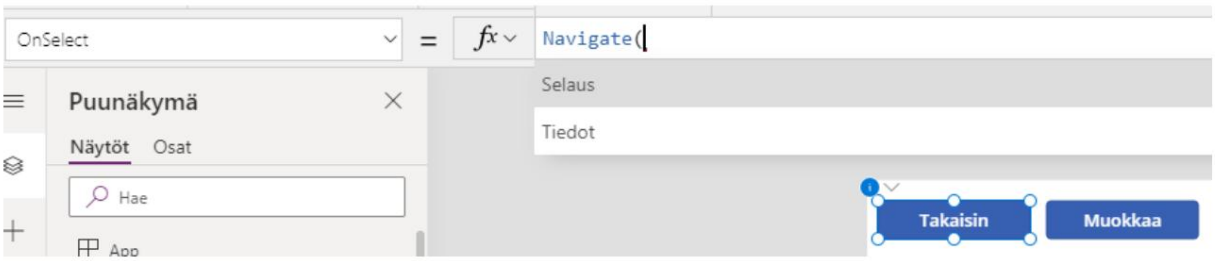
```
Navigate
```

When writing on the formula line, you get a hint of the possible screens where to go (picture 40).

The functionality of the button can be tested by holding **down the Alt key** and clicking

button. The edit screen is not done yet, so can't add a transition to it.

Figure 40 Writing on the formula line



An image can be added to the screen, for example, to improve visibility. Adding a picture

is done by selecting Picture from the Media menu (picture 41). Adding the actual image

takes place through the Picture field in the Properties menu (picture 42).

Figure 41 Image control object

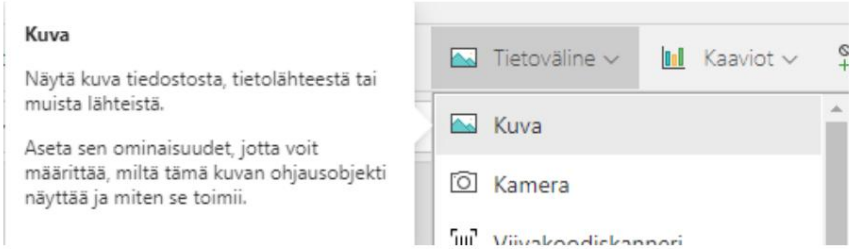
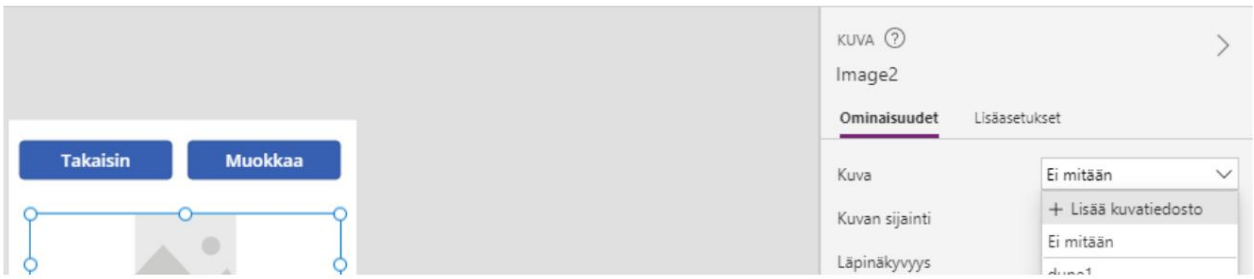


Figure 42 Adding a picture



8.4.6 Editing screen

In the application, you want to edit the data of the selected street name, and for this purpose a new screen is added, what type is edit. Since the street name does not have a lot of information, it is not necessary to scroll screen, so a blank screen is enough. Let's name the new screen **edit.** Let's add a form on the screen (picture 43), connect it to the data source, add fields and move form downwards (picture 44).

Figure 43 Edit the form

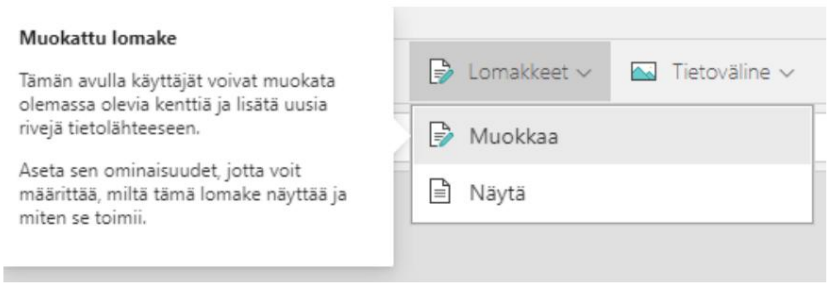
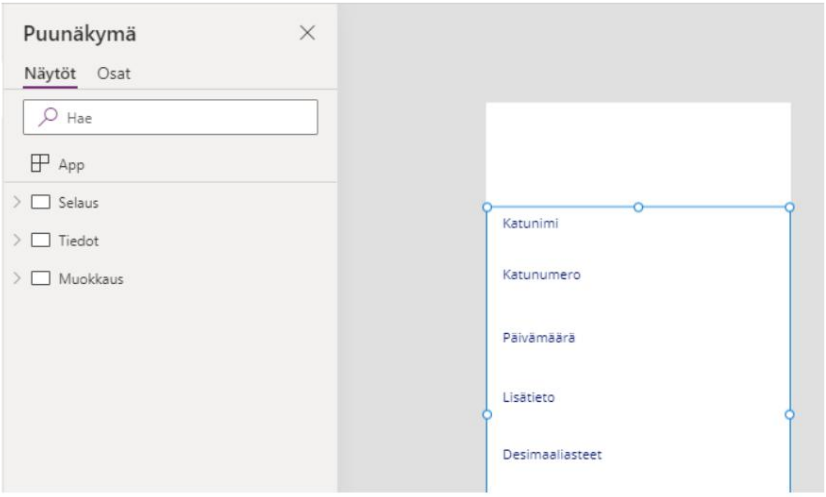


Figure 44 Editing screen with fields



The form on the edit screen does not show the information of the selected street, because two things are missing.

You have to tell the form which street name data you want to edit and on the data screen

you need to add switching to editing mode and switching to the editing screen to the button.

Adding syntax (Syntax 7) to the Item data in the advanced settings for the form on the editing screen.

Syntax 7 Displaying the information of the selected street name in the edit screen

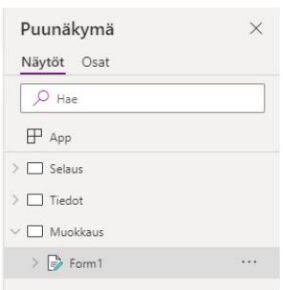
```
BrowseGallery1.Selected
```

Syntax is added to the formula line of the Onselect property for the Edit button on the Details screen (Syntax 8). Form1 is the name of the form on the editing screen (picture 45).

Syntax 8 Switching to editing mode and switching to the editing screen

```
EditForm(Form1);;Navigate(Edit)
```

Figure 45 Contents of the Edit screen in the tree view



To save the changes made and go back to the information screen, add Save-
button and for that OnSelect property the syntax (Syntax 9). Figure 46 shows the Save-
button.

Syntax 9 Save the changed data and go to the data screen

```
SubmitForm(Form1);;Navigate(Data)
```

Figure 46 Save button on the edit screen

8.4.7 Testing

You can test the functionality of the application using the preview function. Testing is performed
in the following manner:

1. Select a street name on the browsing screen, then go to to street information.

2. On the Details screen, press the Edit button, after which the details are
changeable.

3. With the Save button, the data is saved and the changed data can be seen on the data screen

4. Use the back button to go back to the browsing screen.

8.4.8 Recording and Publishing the Application

When the application is ready and you want to "take it into production", you have to publish it. Published
takes place through the File menu (picture 47). After publication, it is distributed to users (Figure 48).

Figure 47 Publication of the application

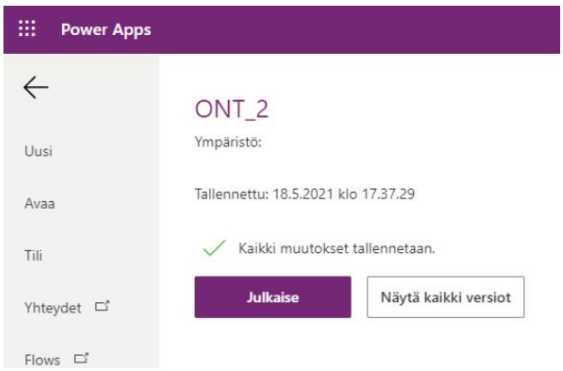
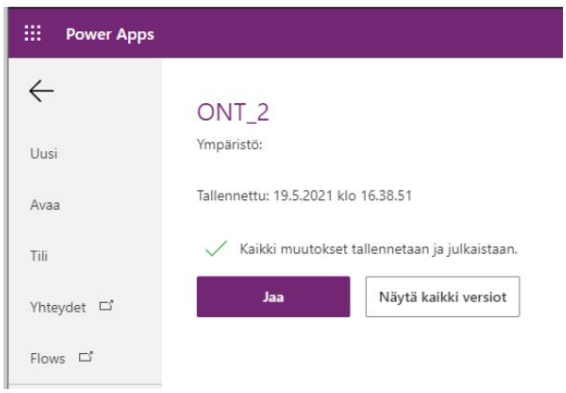


Figure 48 Sharing the published application



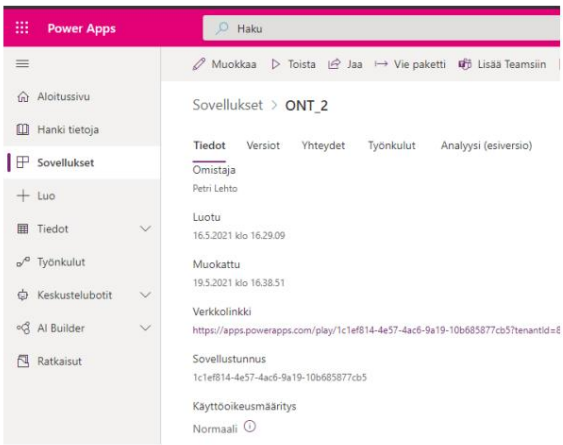
The published application can be developed and changes are only available to users after publication.

8.4.9 Application Information

In the information of the application you can find (picture 49):

- Link to the application
- Versions automatically saved by the system, which can be restored
- Connections used by the application

Figure 49 Application information



9 Conclusions and reflection

With the Power Apps program, you can make different applications and they can be very good complex and contains many different functionalities. However, the purpose of my work was summarizes as well as possible the things that should be adopted first. In the theory part of my work I brought up the most important things and with a simple example application I showed the things in practice in the part.

For the thesis process, I wanted to make a suitable example app to show issues brought up in the theory part. The sample application had to be simple enough, but on the other hand contains the desired things. While making the sample app, I noticed things that it was good to bring up and explain in the theory part.

I think I managed to highlight and narrow down the things that I have noticed required to start using Power Apps.

It's easy to start using Power Apps, but it's worth taking the time to learn it. What the better you know how and what you can do with it, the better application you can make. You should also set aside enough time to make the application. Although Power Apps advertised by the speed of making the application, it takes time to make all applications time. It takes a surprising amount of time to make and finish screens and functionalities time. Very often, estimating the amount of work in advance is very difficult in the early stages and only gets better through routine.

Those who have used Power Apps for a longer time do not necessarily see anything new in the thesis and the work is mainly aimed at new users.

10 Summary

I think I was able to answer the research questions well within the scope of the thesis.
As the work progressed, I learned new ways of using Power Apps, as well as making applications and on the other hand
I reinforced what I had already learned. I gained more confidence in making applications and returned
reminded me of the importance of planning. In the case of new learned things, I had to think,
is the matter in question something that should be brought out in my work.

In my opinion, making good and functional applications with Power Apps requires constant effort
learning and development. Familiarity with existing and new functionalities and
utilization are very important in making applications. Information sharing by users
among is valuable and promotes everyone's learning.

Sources

April Dunnam - YouTube. (n.d.). Retrieved August 6, 2021, from https://www.youtube.com/user/apeyd41686

Learn PowerApps - YouTube. (n.d.). Retrieved August 6, 2021, from https://www.youtube.com/playlist?list=PLcwrIWK7WBcRyFBAFeC-Ws8kJYAGTc8Qi

Microsoft. (nd-a). *Find Features - Power Apps | Microsoft Docs.* Retrieved May 15, 2021, from https://docs.microsoft.com/fi-fi/powerapps/maker/canvas-apps/reference-properties

Microsoft. (nd-b). *Functions, Signals, and Enumerations - Power Apps | Microsoft Docs.* Retrieved April 13, 2021, from https://docs.microsoft.com/fi-fi/powerapps/maker/canvas-apps/formula-reference

Microsoft. (nd-c). *Add and manage connections for chart apps - Power Apps | Microsoft Docs.* Retrieved May 15, 2021, from https://docs.microsoft.com/fi-fi/powerapps/maker/canvas-apps/add-manage-connections

Microsoft. (nd-d). *Getting started with formulas in foundational applications - Power Apps | Microsoft Docs.* Retrieved May 15, 2021, from https://docs.microsoft.com/fi-fi/powerapps/maker/canvas-apps/working-with-formulas

Microsoft. (nd-e). *Save and publish a fabric app - Power Apps | Microsoft Docs.* Retrieved May 18, 2021, from https://docs.microsoft.com/fi-fi/powerapps/maker/canvas-apps/save-publish-app

Microsoft. (nd-f). *Start a workflow in a foundational application - Power Apps | Microsoft Docs.* Retrieved April 25, 2021, from https://docs.microsoft.com/fi-fi/powerapps/maker/canvas-apps/using-logic-flows

Microsoft. (nd-g). *Microsoft Power Apps Help - Power Apps | Microsoft Docs.* Retrieved April 25, 2021, from https://docs.microsoft.com/fi-fi/powerapps/

Microsoft. (nd-h). *Introduction to Power Apps project design - Power Apps | Microsoft Docs.* Retrieved May 15, 2021, from https://docs.microsoft.com/fi-fi/powerapps/guidance/planning/introduction

Microsoft. (nd-i). *About Power Apps Studio - Power Apps | Microsoft Docs.* Retrieved May 18, 2021, from https://docs.microsoft.com/fi-fi/powerapps/teams/understand-power-apps-studio#5--formula-bar

Microsoft Power Apps - YouTube. (n.d.). Retrieved August 6, 2021, from https://www.youtube.com/channel/UCGfWR2ekfRFckLjev6eQYLg

Microsoft PowerApps - YouTube. (n.d.). Retrieved August 6, 2021, from https://www.youtube.com/playlist?list=PLCGGtLsUjhm2bonhBZuEhZU72QkFjOpc6

PowerApps | Microsoft Power Apps - Blog. (n.d.). Retrieved August 6, 2021, from https://powerapps.microsoft.com/en-us/blog/category/uncategorized/

POWERAPPS-related blogs - Blogit.fi. (n.d.). Retrieved August 6, 2021, from https://www.blogit.fi/tag/powerapps

Appendix 1: Material management plan

The thesis is saved on the workstation and there is a backup of it on a separate SSD drive and in WebStorage. When doing the thesis, different versions of it were saved as the work progressed and these is stored on the workstation and a separate SSD drive.

A street name file has been created for the example application, which is saved on the workstation and to OneDrive. The street name material has been copied from the internet and is open data.

The example application has been made using the employer's Power Apps and the application has been saved to the employer's environment. A package is taken from the example application after the work is completed, which stored on a personal workstation and a separate SSD drive.

Appendix 2: Pictures of the example application

The three screens of the sample application are browsing, information and editing.