Expertise
and insight
for the future

Marko Kalevo

Building a Graphical User Interface with MATLAB

Metropolia University of Applied Sciences

Bachelor of Engineering

Industrial Management

Bachelor's Thesis

13 November 2019

Author Title Number of Pages Date	Marko Kalevo Building a Graphical User Interface with MATLAB 54 pages 13 November 2019
Degree	Bachelor of Engineering
Degree Programme	Industrial Management
Professional Major	Process Management
Instructor	Nina Hellman, Head of Degree Programme, Senior Lecturer

The subject of this Thesis is to build a Graphical User Interface with MATLAB computer software which supports the creation of air quality reports. The Thesis was carried out for The Ministry of The Environment of Chile and to solve a real-world problem.

The main goal was to create an easy-to-use graphical user interface which does not require advanced technical knowledge to use. The aim was to improve the working process and make it possible to carry out the task of making reports without having technical knowledge of using MATLAB. The general requirements for the User Interface were functionality, clarity and Ministerial design. The Graphical User Interface created in this thesis unites two MATLAB programs under one application where the variables are easy to choose for executing the program.

This Thesis explains the entire project of building the graphical user interface from planning to usability inspection analysis of the ready-made standalone application. It starts with examining the problem and planning the project and proceeds to exploring the needed tools and to the building of the application itself. The last step of the project was conducting a Heuristic Evaluation for the application which was performed in order to find possible usability problems in the application.

The Thesis explored MATLAB and its possibilities in building User Interfaces and standalone applications. The second benefit of this Thesis was to get the contact person of the customer acquainted with the building of graphical user interfaces with MATLAB. In general, this Thesis can be useful for a reader who is interested in learning about the building of GUI’s and the process itself, building GUI’s with MATLAB and for anyone interested in conducting a Heuristic Evaluation.

Keywords	Graphical User Interface, GUI, MATLAB, Heuristic Evaluation

Tekijä Otsikko Sivumäärä Aika	Marko Kalevo Graafisen käyttöliittymän luominen MATLAB:lla 54 sivua 13.11.2019
Tutkinto	Insinööri (AMK)
Tutkinto-ohjelma	Tuotantotalous
Ammatillinen pääaine	Teollisuuden Prosessit
Ohjaajat	Nina Hellman, Tuotantotalouden tutkintovastaava, Lehtori

Tämän insinöörityön aiheena on rakentaa graafinen käyttöliittymä MATLAB-ohjelmalla, joka tukee ilmanlaadun raporttien luomista. Insinöörityön tuotos on tehty todelliseen tarpeeseen Chilen Ympäristöministeriölle ja ratkaisemaan todellista ongelmaa.

Insinöörityön päätavoitteena on luoda helposti käytettävä graafinen käyttöliittymä, jonka käyttäminen ei vaadi korkeaa teknistä osaamista. Käyttöliittymän tavoitteena on parantaa työskentelyprosessia sekä mahdollistaa ohjelman käyttäminen muilta kuin itse pääkäyttäjältä. Käyttöliittymän avulla on tarkoitus yhdistää kaksi MATLAB-ohjelmaa yhden ohjelman alle sekä luoda visuaalinen käyttöliittymä, josta voidaan valita ohjelman suorittamiseen vaadittavat muuttujat. Käyttöliittymän yleiset vaatimukset ovat: toimivuus, selkeys, helppokäyttöisyys sekä ministeriöllinen ulkonäkö.

Insinöörityö käsittelee koko graafisen käyttöliittymän rakentamiseen kuuluvaa prosessia suunnittelusta valmiin sovelluksen käytettävyys-analyysiin. Se alkaa yleisesti ongelman tarkastelemisella ja projektin suunnittelulla, josta se etenee työkaluihin tutustumiseen sekä itse sovelluksen toteuttamiseen. Viimeisenä vaiheena valmiina oleva sovellus tutkitaan Heuristisella arvioinnilla, jonka tarkoituksena on löytää mahdollisia käytettävyysongelmia sovelluksesta.

Insinöörityö tutustuu MATLAB:iin ja sen mahdollisuuksiin käyttöliittymien sekä sovelluksien rakentamisessa. Insinöörityön toisena hyötynä valmiin sovelluksen lisäksi oli tutustuttaa asiakkaan työntekijä käyttöliittymien luomiseen. Yleisesti tämä insinöörityö voi olla hyödyllinen lukijalle, joka haluaa tutustua graafisen käyttöliittymän rakentamiseen sekä sen prosessiin, rakentaa graafisen käyttöliittymän MATLAB:lla tai Heuristisen arvioinnin toteuttamisesta kiinnostuneelle henkilölle.

Avainsanat	MATLAB, Graafinen Käyttöliittymä, Heuristinen arviointi

Contents

List of Abbreviations

MATLAB Computer software for different calculations and its code language. Used in math-related tasks.

MMA Ministerio del Medio Ambiente, the customer of the project. In English; The Ministry of The Environment of Chile.

GUI Graphical User Interface. An interface to communicate with the machine.

GUIDE GUI development environment, an environment of graphical user interface development in MATLAB computer software

STEM An acronym for the fields of science, technology, engineering and mathematics.

M-file The normal file of MATLAB. File format .m.

F-file The figure file which MATLAB uses in making of Graphical User Interfaces with GUIDE. File format .fig.

Callback Function pointer to another function to communication with the concerned user interface control. A function used in MATLAB GUI’s.

.exe An executable program. The standard file extension used by Windows programs.

1 Introduction

Graphical User Interfaces, often shortened as GUI, is an interface to the user which helps communicating with the machine. GUIs offers visual support to the users and basically an environment, where the user is familiar without prior knowledge. Well-designed Graphical User Interface makes operation with the machine more intuitive and user-friendly. Graphical User Interfaces with a good workflow improve the working process of organizations tasks and can be used by almost anybody.

The objective of the project is to make an application for The Division of Air Quality at The Ministry of The Environment of Chile, which helps the workers to perform daily routine task without specific technical knowledge. The daily routine task is to simulate reports about the quality of the air. The reports are simulated with a program which is executed in MATLAB computer software.

The goal of the project is to make Graphical User Interface with MATLAB computer software for two already existing programs which makes reports of air quality. The intended outcome is a standalone application, which is easy to use and have a good user experience. The main benefits of the application are to unite two programs under one, share the task to generate reports with other persons in the office and improve the workflow. The other advantage of the project is getting the contact person of the customer acquainted with the possibilities of making user interfaces with MATLAB.

A Graphical User Interface is a visual helper to the user to communicate with the machine. This thesis contains familiarizing with MATLAB-software, investigation of the different ways of building graphical user interfaces with MATLAB, designing of user interface, building and programming the application with MATLAB with the management of the project. Also, in the end the application is evaluated with a Heuristic Evaluation.

The project came up from visiting in The Ministry of The Environment of Chile, by asking the possibilities of making the thesis project for them. The explained problem as also the topic of this thesis, was to build a solution for a simple daily task, which needed specific knowledge about MATLAB environment. Because of the specific needed knowledge, the task is only maintained by two persons. The expected benefits of the project are to have wider scale of the users and make the task easier to manage.

The project started by analysing the current state by inspecting the problem with the current situation and benefits of the building a graphical user interface over the already existing programs. From that, the project proceeded with different steps to the standalone application.

2 Project management & Planning

The project got started by visiting The Air Quality Division of The Ministry of The Environment of Chile (Ministerio del Medio Ambiente, MMA). In the Air Quality Division, they investigate the air quality and pollution in the Chile and make analyses from the current air data. Their role is to share the information with other stakeholders and publish the information on their website and other platforms.

In The Ministry of The Environment they use the computer software MATLAB as a part of their information systems for making analyses, calculations and visualizes from the different data about the weather. Concerning to the task, everyone in the office does not have the ability to use MATLAB. The Graphical User Interface would be an important help to the office by helping the employees to execute a daily routine task without specific knowledge of using MATLAB.

The purpose of the project is to create a Graphical User Interface using MATLAB-software. The client customer of the thesis already has the program, which is making the calculations, but without a Graphical User Interface. Currently there are two programs because there are two different variations which both need their own program. By running the programs, the programs make reports about the selected day or month (depended which program is used). The programs use measured data of the air quality and simulates the data for a published form, to PDF and into their internal web platform. Because the current versions of the executable programs are already made, the project focus for making the front-end interface to unite them and not exactly for the original programs.

2.1 Background

Air quality is a problem in many cities in Chile. The Ministry of The Environment of Chile is responsible of the informing and publishing information about the air quality. The quality of the air needs to be known from each day and be published in their platforms and to their stakeholders. Especially in wintertime, it is higher probability that the quality of the air is not that good comparing to summertime. In wintertime there is less wind and people use unclear heating methods to warm their houses, like gas and firewood. Chile is a country which have problems in the air pollution, especially in the bigger cities. The mountain line called Andes are like a wall for the thin and long Chile. It blocks the air and do not let the wind move the contamination away over the cities (especially in the capital, Santiago). Air pollution is also big cost for health sector, making at least 670 million dollars cost, causes 127 000 emergency health consultations and more than 4 000 premature deaths. (UN Environment 2017)

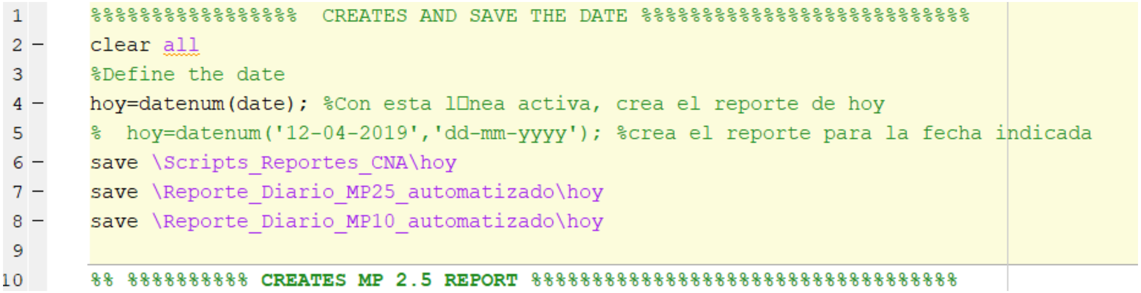
Image 1. Beginning of the winter in Santiago de Chile, 3.6.2019. The air pollution is easy to see with eyes.

Air pollution is measured with values PM2.5 and PM10, where the “PM” means the “particulate matter” and number means the size of the particulate matter. PM2.5 means the particulate matter with the size of 1.0 – 2.5 microns, which is about 3 % of the diameter of a human hair. These PM2.5 are formed from traffic, industry, wood burning, smoking and energy production. PM10, the particulates with a bit more size, forms for example from the soil, road elapse, sand and spring dust. PM2.5 are more dangerous of these two, because of their small size they easier get deeper into the human body than PM10. Briefly explained, PM10 effects for respiratory tract and PM2.5 for lungs and blood circulation. (Jussi Huttunen 2012)

2.2 Current state analysis

In the Division of the Air Quality, the workers use MATLAB to make analyses about the air quality. Concerning the thesis, in MATLAB they have programs, which make analyses about the data of air pollution and publish the output in readable form. The output of the program is published in their platform for their stakeholders. The output is shown in an internal website and is possible to download from there as PDF document, which is made by the same MATLAB program.

In the division of the air quality, the workers use two different programs to make the reports. The other program is to make daily reports of the air pollution and the other makes monthly. Daily reports are used to follow current or exact day situation of the air quality. Monthly reports are used to investigate the air quality in long-term by investigating the averages of the months and comparing them to earlier years. The reports are used for publishing the information for citizens, stakeholders and to maintain better air quality in the country. The programs read the air pollution data from the datastores, which comes from the measurement centres. The programs visualize the current information of the month or inputted date from data to the user-friendly form with diagrams, charts, numbers and colours.



```
1      %%%%%%%%%%%%%%%%%  CREATES AND SAVE THE DATE %%%%%%%%%%%%%%%%%%%%%%%%%%%
2 –    clear all
3      %Define the date
4 –    hoy=datenum(date); %Con esta l□nea activa, crea el reporte de hoy
5      %  hoy=datenum('12-04-2019','dd-mm-yyyy'); %crea el reporte para la fecha indicada
6 –    save \Scripts_Reportes_CNA\hoy
7 –    save \Reporte_Diario_MP25_automatizado\hoy
8 –    save \Reporte_Diario_MP10_automatizado\hoy
9
10     %% %%%%%%%%%% CREATES MP 2.5 REPORT %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Image 2. The first rows of the daily report program.

From the image above is shown the first lines of the code of the original program which executes the daily reports. Texts in green colour are comments and the rest are the code. The lines 4 and 5 are for the variable day (var hoy), where the user makes changes depending on which day document is needed. In line 4, it is possible to run documents from the current date and from the line 5, it is possible to choose the date by writing the wanted day to the code line. As this program is for running daily reports, there is also an individual program to make monthly reports. The first lines of the monthly report are similar except the difference of the save folder and the difference for choosing day, the month is chosen. The month is chosen by choosing the first day of the next month.

Basically, every time when running the program, it is needed to choose the correct program (daily or monthly) and make the changes for the date variable depending which day or month is needed or wanted to print out. For the frequent MATLAB user this task is simple, but for the person who do not have knowledge about MATLAB or a person with a low information technology knowledge the task of running these documents is difficult. Because of that, there has been a problem to maintain this task when the responsible person of this task has not been working and the task hasn't been performed.

Also, problem with the current version is that the programs automatically runs the documents about both values, PM2.5 and PM10. That is not always necessary and wastes time. Running the program takes approximately 8-10 minutes and the time is shared by each report.

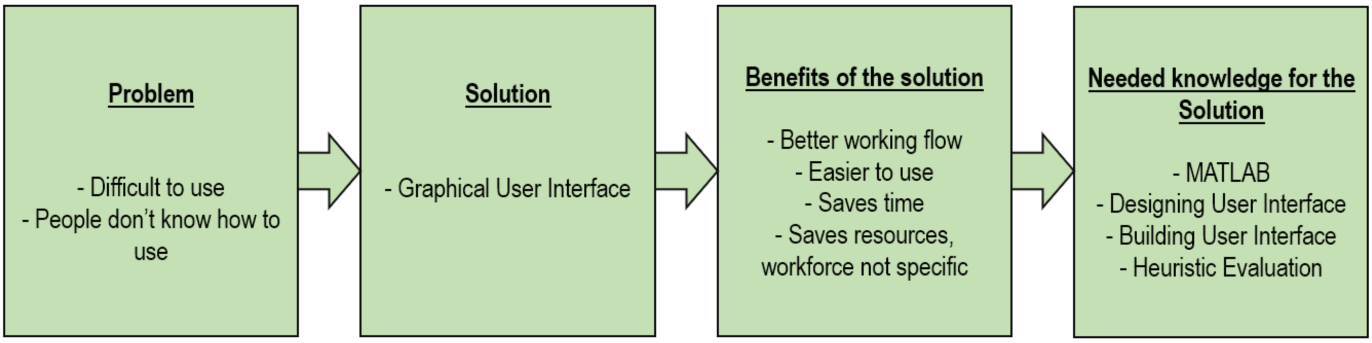

Problem
- Difficult to use
- People don't know how to use
Solution
- Graphical User Interface
Benefits of the solution
- Better working flow
- Easier to use
- Saves time
- Saves resources, workforce not specific
Needed knowledge for the Solution
- MATLAB
- Designing User Interface
- Building User Interface
- Heuristic Evaluation

Image 3. Current problem, solution, benefits of the solution and needed knowledge to implement the solution.

2.3 Goals & Requirements

The goal of the project is to build a standalone application. The application will be built over already existing MATLAB programs, which simulates the reports of the air quality. The application helps the workers to maintain a daily executed task and makes it possible for every person in the office to execute the task without technical knowledge.

In the current version it is needed to open the correct program (daily or monthly) with MATLAB and select the date from the code editor. That needs a bit technical knowledge and there are only two persons in the office who manages the job. That means, if both two persons are in holidays or sick leaves the task will not get carried out. So, the scope of the project is to make an individual standalone application (.exe), where the variables are easy to choose from Graphical User Interface components. The standalone application would make possible to handle the task by more people of the office with the benefit of saving a time in the selecting of the correct program and variables.

The other desired benefit of the application is better user experience in executing the task. With a graphical user interface would be possible to choose the report, daily or monthly and the variables of the air quality, PM2.5, PM10 or both. Earlier the daily and the monthly report have been executed from different programs and the air quality variable haven't been possible to change. Also, the date has been inputted manually inside the program but with an application it will be chosen from the graphical user interface components. These are the main requirements of the project:

- Building of graphic user interface for existing program, opened in MATLAB or standalone .exe. Prefer standalone desktop application.

- Good workflow in the application.

- Possible to choose between variables daily or monthly report and choose the date for them. Also, possible to choose the variable's PM2.5, PM10 or both.

- Designed with the colours of The Ministry of The Environment including the logo (visual appearance).

When the requirements of the project are met, the benefits of it should become clear. The GUI of the program will give better accuracy, efficiency and consistency. These benefits should be seen in a bigger number of users due to easier workflow. The benefits can also be inspected by process and the changes in it. In Image 4, the current process of the executing is compared with the process with the application.

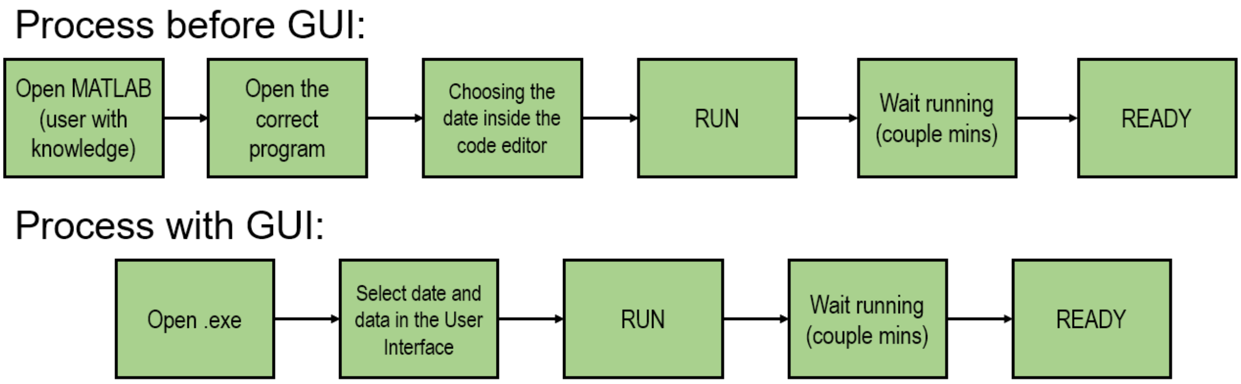

Process before GUI:
Open MATLAB (user with knowledge)
Open the correct program
Choosing the date inside the code editor
RUN
Wait running (couple mins)
READY
Process with GUI:
Open .exe
Select date and data in the User Interface
RUN
Wait running (couple mins)
READY

Image 4. Process to print reports. Before and after a graphical user interface.

The change in the process is not very big when examining the new process for the earlier one. Also, generating the reports with the new interface does not really save much in time, excepts when printing only the other of the variables PM2.5 or PM10. The main difference with the process is, that in the current method its needed to open the correct program with MATLAB and in the new solution, both reports are possible to generate under one program. Also, the GUI will give flexibility with the choosing of the values PM2.5, PM10 or both. The main value which comes with the new solution is; it does not require professional knowledge in using and being acquainted with the MATLAB environment to make date changes in the code editor.

2.4 Steps of the Project

The project started from the visit of the office of The Ministry of The Environment of Chile on 3 April 2019. In that time, it started roughly by visualizing the project in the big picture and considering its suitability for a thesis topic taking into account scope and depth. After getting acceptance from the teacher it got under the work.

Building a Graphical User Interface needs specific knowledge about making interfaces and especially in this case, about making them with MATLAB. The project begins with determining the problem, planning, and current state analysis, to understand why the application is to be built and understand the benefits. Next the focus is on “how it is built” by familiarizing MATLAB computer software and its possibilities to make GUI’s with the basics of designing user interfaces and experience. Also, in the building of an application it is important to inspect that the product is good in the end by testing it with users. The project followed the process seen in the figure below.

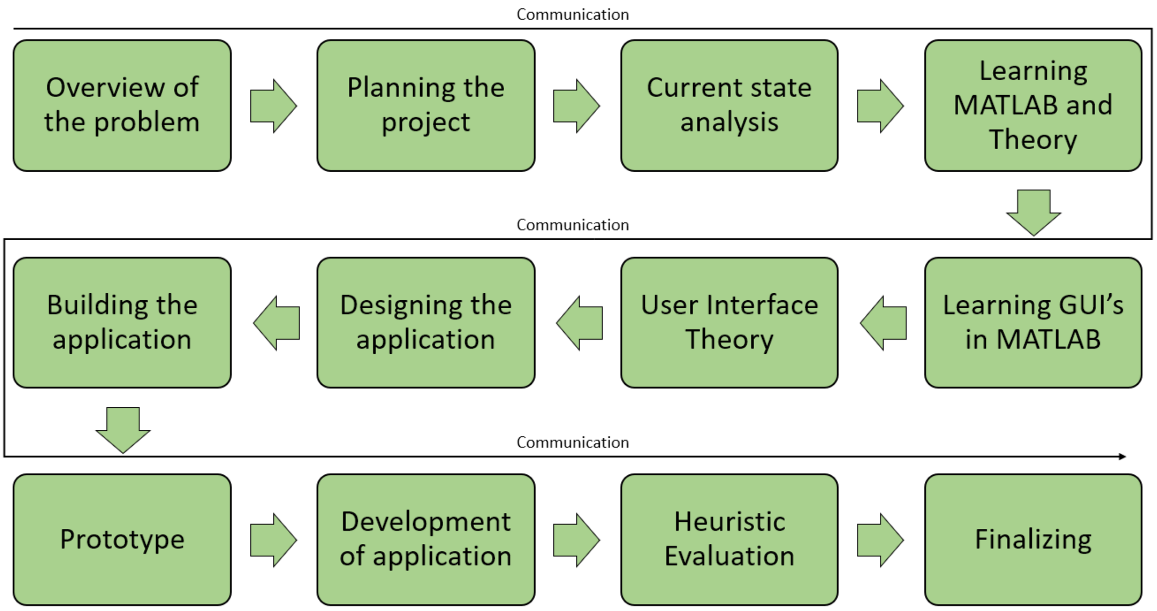
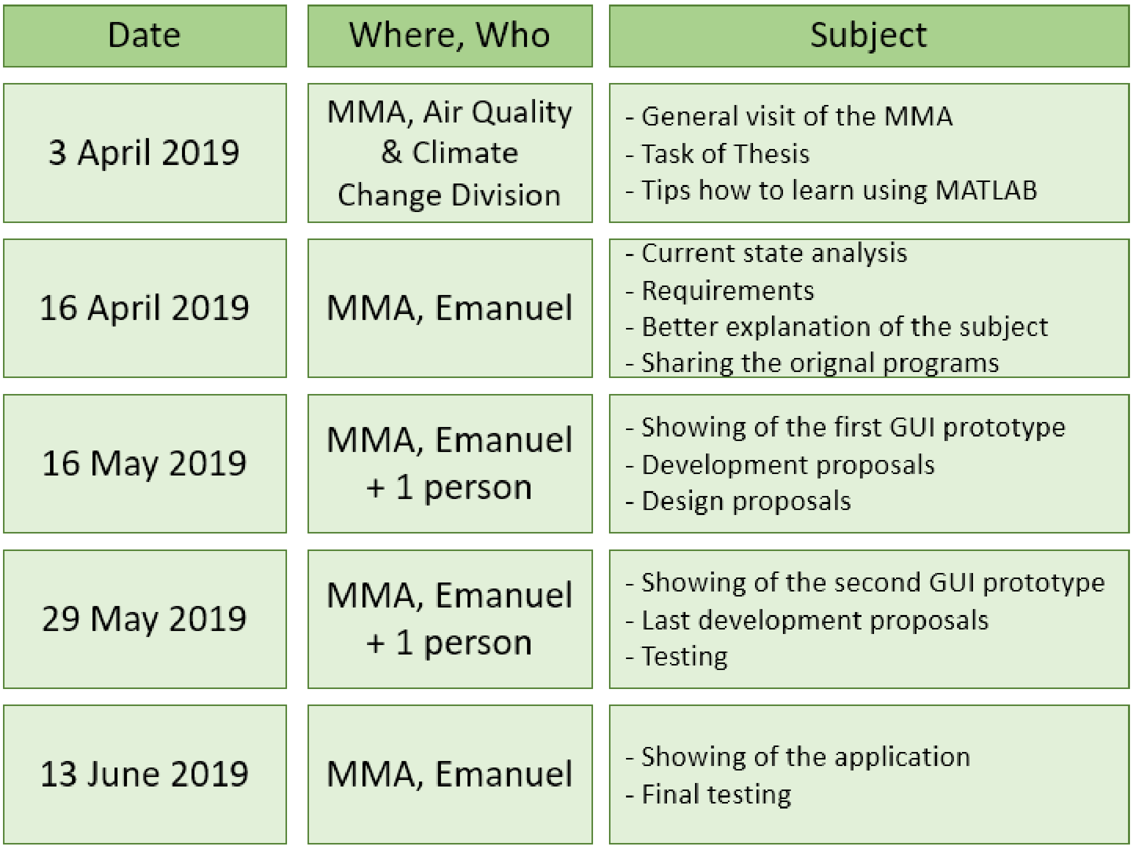


Image 5. Working process of the project.

This project needs different knowledge for carrying it out commendably as seen in the Image 5. Basically, it is not only about the using and making interfaces with MATLAB. The important skills of this project consist from project management, communication, designing user interfaces and experience, technical knowledge and development of the application.

The customer is met five times in the project. The first meeting is a general visit where my missing thesis subject came to the subject and there was found a suitable project for it, building a graphical user interface for the air quality analysis programs which are operated with MATLAB.

As the programs are operated with MATLAB-software and being the main tool of the thesis, the project started by practicing of the using of it. After that, the meetings are settled via WhatsApp and E-mail with the contact person of the customer, Emanuel.



Date	Where, Who	Subject
3 April 2019	MMA, Air Quality & Climate Change Division	- General visit of the MMA - Task of Thesis - Tips how to learn using MATLAB
16 April 2019	MMA, Emanuel	- Current state analysis - Requirements - Better explanation of the subject - Sharing the orignal programs
16 May 2019	MMA, Emanuel + 1 person	- Showing of the first GUI prototype - Development proposals - Design proposals
29 May 2019	MMA, Emanuel + 1 person	- Showing of the second GUI prototype - Last development proposals - Testing
13 June 2019	MMA, Emanuel	- Showing of the application - Final testing

Image 6. Meetings with the customer and the subjects of the meetings.

The second meeting focused on better explanation of the project, requirements and to the current state analysis. The task was defined more detailed to be possible to start solving it. After the current state analysis, next step was focusing to the different methods of building graphical user interfaces with MATLAB and to choose the correct one for the project.

One month later the different methods of building GUI's with MATLAB were researched and the first prototype of the graphical user interface ready. The prototype was inspected by the contact person and the list of development proposals done. In the 4th meeting the graphical user interface was ready to the testing even having some small issues.

Before the final meeting with the customer, the last problems of the GUI were solved, and the GUI compiled to the application. The application is developed iteratively by gathering the new needed information for the problems which were encountered in the application development.

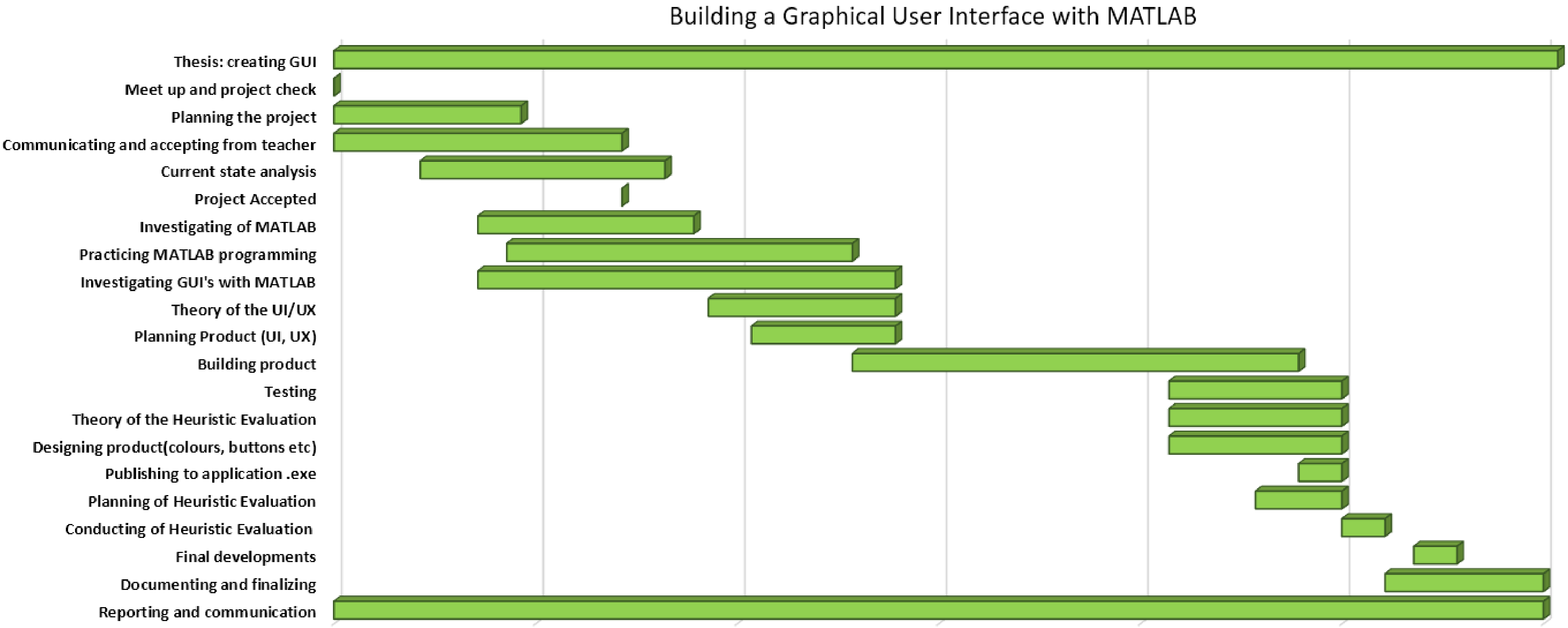


Image 7. The building process shown in the simplified Gantt chart.

3 Tools and Theory

This chapter focuses for the necessary tools and theory which were needed in creating the graphical user interface for this project. It begins from the main needed software of this subject and how the interfaces are possible to make with it. That is MATLAB, the computer software which is better explained in the next subsection. But shortly, it is not originally a program to build individual applications but has possibility for it. MATLAB is a calculator with a programming language, which performs in different math-related tasks. The software has three different ways to build applications, which are tested and compared to find the best one for this project.

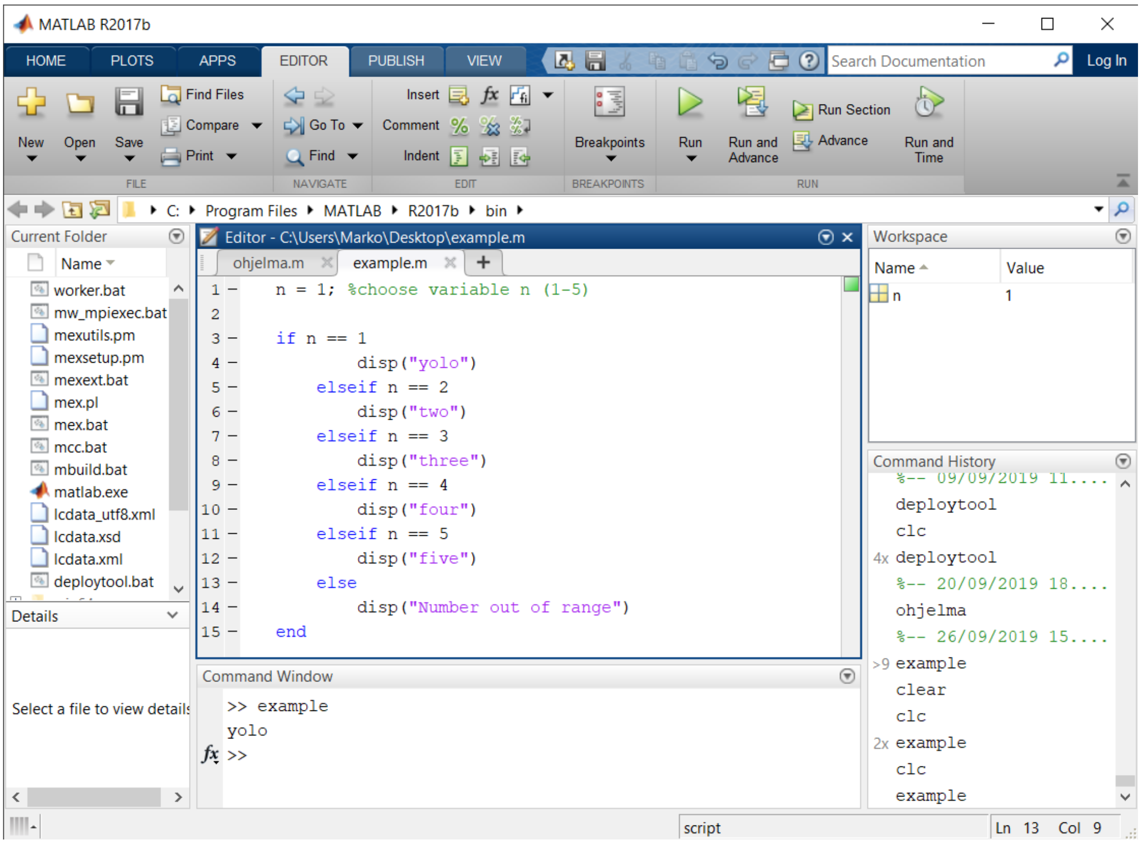
Creating a good user interface is not totally about the technical knowledge. It is important that the application would also be user-friendly. The good User Interface and Experience (UI and UX) have raised probably one of the most important competition advantages in the last years. The application of the thesis itself does not have literally a market value; the value is measured with better process in the office work. With a good design, UI and UX, the application will fulfil better the user's need.

The last subsection examines a method for finding usability problems. The method is called Heuristic Evaluation and it is used usually when the prototype of the application is ready. The method is for finding usability problems which would probably later be occurred by the end-user. Subsection 3.4 introduces the process and instructions of Heuristic evaluation. It is based on the "Chapter 1. Heuristic Evaluation" of the book "User Interface Inspection Methods" which is made by Chauncey Wilson.

3.1 MATLAB in general

MATLAB (matric laboratory) is a computer software, as also called the programming language which the software uses. It is multi-paradigm numerical computing environment developed by MathWorks. MATLAB is a calculator with a programming language, which performs in different math-related tasks. The common math-related tasks where MATLAB performs are Numerical computation, Visualization and Programming. (Sizemore and Mueller, 2014, Chapter 1)

The use of MATLAB can be wide, from simple tasks to complex models. The structure of the language is made simple to help users to focus on their work rather than the complexity, which also means it is less flexible than some other languages. The programming language of MATLAB is called as fourth-generation language (as for example Structured Query Language, SQL), which vernacularly means it is made to remind part of the human language and made possible to follow it without knowing the syntax of the language. The language is designed to focus more to solve real-world problems to professionals of different subjects than being a complex language which needs deep specialization. Because the purpose of the language is to be practical and productive problem solver, the main users of the MATLAB are Scientists, Engineers, Mathematicians, Students, Teachers, Professors, Statisticians, Control technology, Image-processing researchers and Simulation users. (Sizemore and Mueller, 2014, Chapter 1)



```
n = 1; %choose variable n (1-5)

if n == 1
        disp("yolo")
    elseif n == 2
        disp("two")
    elseif n == 3
        disp("three")
    elseif n == 4
        disp("four")
    elseif n == 5
        disp("five")
    else
        disp("Number out of range")
end
```

Name	Value
n	1

```
>> example
yolo
fx >>
```

```
  %-- 09/09/2019 11:...
  deploytool
  clc
4x deploytool
  %-- 20/09/2019 18:...
  ohjelma
  %-- 26/09/2019 15:...
>9 example
  clear
  clc
2x example
  clc
  example
```

Image 8. MATLAB working environment with simple if/else statement.

In the Image 8. is shown the working environment of MATLAB and exactly an editor environment. It contains the editor for code, workspace to show variables, commands and its history with other necessary software components. As MATLAB can be used widely for different needs, here is explained 5 common use of MATLAB:

1. Algorithms and Applications development

The software provides tools to develop, utilize algorithms and applications. With vector and matrix operations, it is used to solve STEM (Science, Technology, Engineering and Mathematics) problems. In the programming language is not necessary to specify data types and allocate memory which makes it faster to develop algorithms than traditional languages. As an Array programming language, one-line of code can replace several line of traditional code languages (for example, C, C++, Java) and it still offers all the features like arithmetic operators, control flow, data structures, data types, object-oriented programming and debugging. (Sizemore and Mueller, 2014, Chapter 1)

MATLAB contains development tools as step by step simulations and code checker, to analyse the code and its functionality. Also, it is possible to record the time of code executes and to do Directory Reports, which finds the differences and dependencies of the files, and the efficiency and coverage of the code. In addition, with MATLAB is possible to make Graphical User Interfaces. GUI's are possible to make programmatically, with GUIDE (Graphical User Interface Developer Environment) and with the newest method, App Designer environment (published in 2016a). GUI's can be used to show up your algorithms and to get better User Experience. GUI's are also possible to deploy for standalone applications with *deploytool*. As building a GUI with MATLAB is a subject of this thesis, the theory contains own chapter for it (Chapter 3.2). (Sizemore and Mueller, 2014, Chapter 1)

2. Data access and Analysis

In terms of data, MATLAB supports with the whole process from the acquisition of data from external devices and databases, with pre-processing, visualization and numerical analysis. With the command line and the different tools of MATLAB, it is a high-level tool for investigating data. MATLAB can read the data from all the most known formats from Excel files to images and sounds. Also, there is possibility to use different programming languages and applications via MATLAB. (Sizemore and Mueller, 2014, Chapter 1)

3. Data visualization

As MATLAB is good to handle data, it offers all necessary functions to visualize data professionally. MATLAB can understand large amounts of data and display it as two-dimensional arrays or three-dimensional scalar or vector data. Two-dimensional diagrams can be viewed from different diagrams and histograms to the animations, as three-dimensional have also their variety of presentations. The visualizations can be exported to the other applications or common graphic formats as GIF, JPEG, BMP, EPS, TIFF, PNG, HDF, AVI, and PCX. It is possible to create and apply style templates to design the exported visualizations with different colours, lines and other visual requirements. (Sizemore and Mueller, 2014, Chapter 1)

4. Numeric Calculation

MATLAB is a calculator and fountained for math-related tasks. It includes loads of functions to mathematic, statistic and engineering fields. Some example of numerical calculations are manipulation of matrices and linear algebra, polynomials and interpolation, statistics and data analysis, optimization and numerical integration and much more. (Sizemore and Mueller, 2014, Chapter 1)

5. Results publication and Applications distribution

The results and works from MATLAB can be published and used between different programs as for example Microsoft Word or PowerPoint. From MATLAB Editor, it is possible to publish automatically code to HTML, Word, LaTeX and other formats. Also, the code can be integrated to other languages and applications. (Sizemore and Mueller, 2014, Chapter 1)

3.2 Graphical user interfaces with MATLAB

User Interface is a tool which helps human communicate with the machine. It is the visually seen part, interface to user to communicate, read and send commands for the technology or a machine (processor) which exists behind the surface. With a user interface, the computer and user exchange information and instructions. In early days of computers was common to interact with the computer trough command line. Nowadays it is basically hidden behind the User Interfaces. (BBC Bitesize: User Interfaces)

```
Compusoft <R> MS DOS
<C> Copyright Compusof
C:\>_
```

File
Edit
View
Bookmarks
TRAVEL
Home
PLACES TO VISIT

Image 9. Without and with user interface (source: BBC, Bitesize.)

With User Interfaces the tasks are performed via Interface and the user does not have to type commands to the command line or create scripts. The end-user neither do not have to understand how the tasks are performed and details of the code which exist behind the interface. Basically, every program in your computer, mobile phone and every web page in a web browser are displayed to you by User Interface. (MathWorks: Creating Graphical User Interfaces 2015, pp. 1-2)

The idea of graphical user interface is to help the user to work, by hiding the needed commands for the operation behind the surface and help the using of the program by serving “easy-to-use” interface between the program and the user. With MATLAB it is possible to do interfaces with different type of basic components. Components can be used to choose variables and for executing the program. Interface offers a way to use a system without technological knowledge. (Eriksson et al, 2004, p. 96)

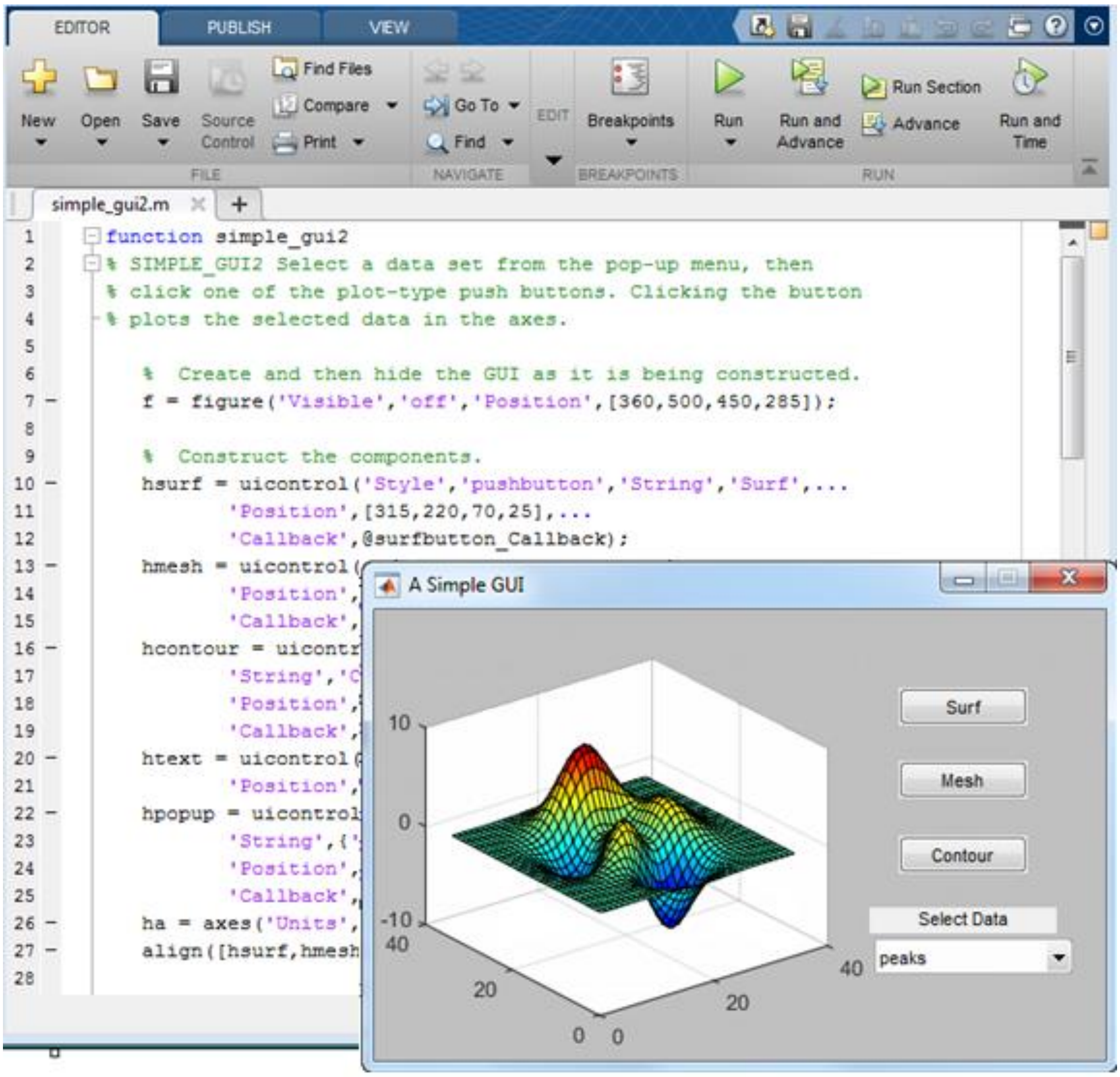
In MATLAB, there are three different ways to make user interfaces. The methods of making Graphical User Interfaces with MATLAB have changed in the developments of the program itself. Those three different ways are programmatically, with GUIDE (Graphical User Interface Development Environment) and App Designer. Each of these methods to build apps have their small different in functionalities and in the process of building. The choosing of the method may be depending of the needs in the project or the preference of the working. (MathWorks: Ways to build Apps 2019)

The using of the different methods is learned mainly from the MathWorks website and from the videos of “MATLAB App Designing: The ultimate Guide for MATLAB Apps” made by Nouman Azam.

3.2.1 Programmatically

By making application programmatically means making layout and the functionalities of the application entirely with MATLAB functions in the code editor. Making application programmatically starts by making a function and creating *figure* for making the figure window in it. Figure means the window which opens as an application or another output as a window. The name, colour, size and other functionalities are possible to modify in the *figure* function. (MathWorks: Create a simple app programmatically 2019)

After having a figure, it is possible to start adding different components into the figure as axes, panels, tables and controls. For example, the controls such as different buttons, checkboxes, radio buttons, slider, pop-up menus and texts are done with a function *uicontrol*, meaning user interface control. The style, positions and values are set in the function as also the name of the C*allback*, which is a function pointer to another function to communication with the concerned user interface control. (MathWorks: Create a simple app programmatically 2019)



```
function simple_gui2
% SIMPLE_GUI2 Select a data set from the pop-up menu, then
% click one of the plot-type push buttons. Clicking the button
% plots the selected data in the axes.

   %  Create and then hide the GUI as it is being constructed.
   f = figure('Visible','off','Position',[360,500,450,285]);

   %  Construct the components.
   hsurf = uicontrol('Style','pushbutton','String','Surf',...
          'Position',[315,220,70,25],...
          'Callback',@surfbutton_Callback);
   hmesh = uicontrol(
          'Position',
          'Callback',
   hcontour = uicontr
          'String','C
          'Position',
          'Callback',
   htext = uicontrol(
          'Position','
   hpopup = uicontrol
          'String',{'
          'Position',
          'Callback',
   ha = axes('Units',
   align([hsurf,hmesh
```

Image 10. Matlab gui programmatically. (Source: MathWorks.)

When working programmatically, the application is form under one file. That file is M-file (file mode .m), which is the basic MATLAB program file mode. That gives the best advantage in flexibility, to interact with other MATLAB program comparing for the two next shown methods for building Graphical User Interfaces.

3.2.2 GUIDE

MATLAB has an own Graphical User Interface Development Environment called GUIDE, which allows the user to make, design and edit user interfaces. In MATLAB, by entering *guide* into the command window, it opens the window to open existing GUI or create a new one. In GUIDE, it is possible to build interfaces with good variation of basic components of the graphical user interfaces. By comparing to the programmatical way it is easier to add the buttons and do the layout of the window. When making graphical user interface with GUIDE, the program consists from two files, a normal MATLAB file (M-file, .m) and a figure file (F-file, .fig), which is the file of the figure of the graphical user interface. (Nouman Azam 2016)

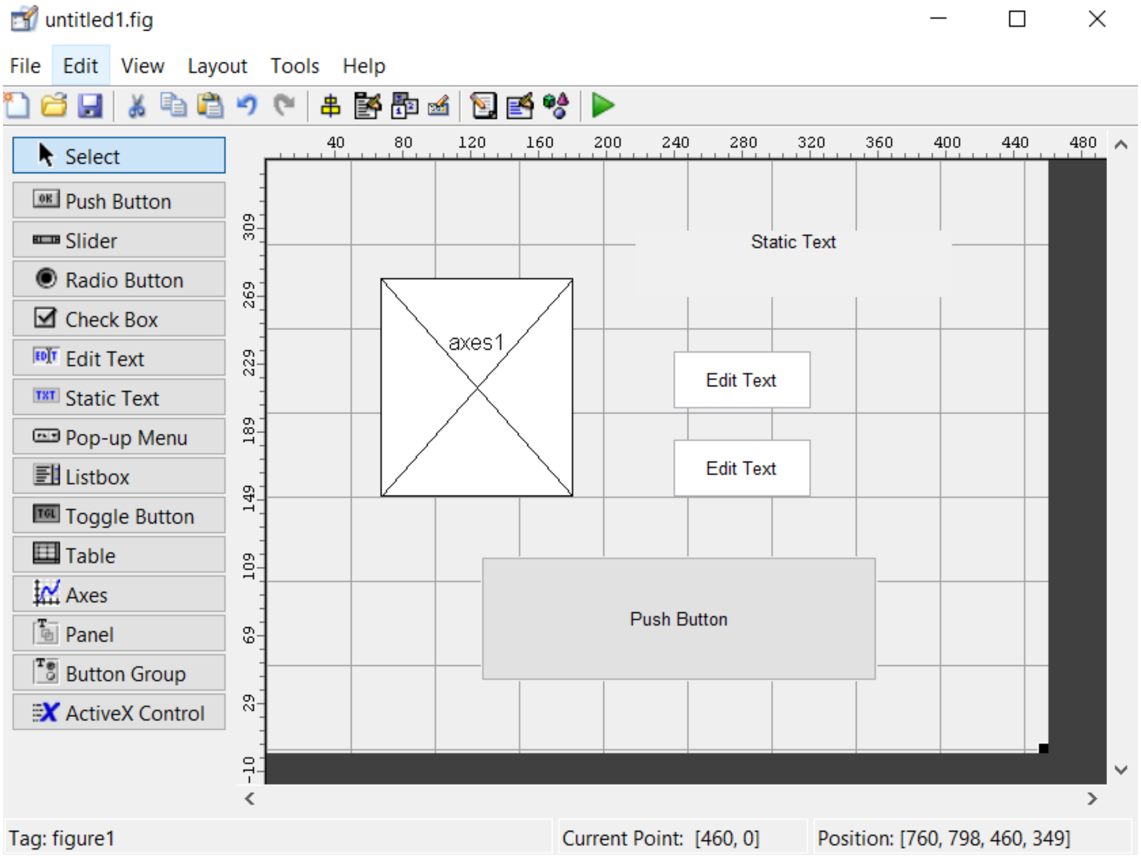
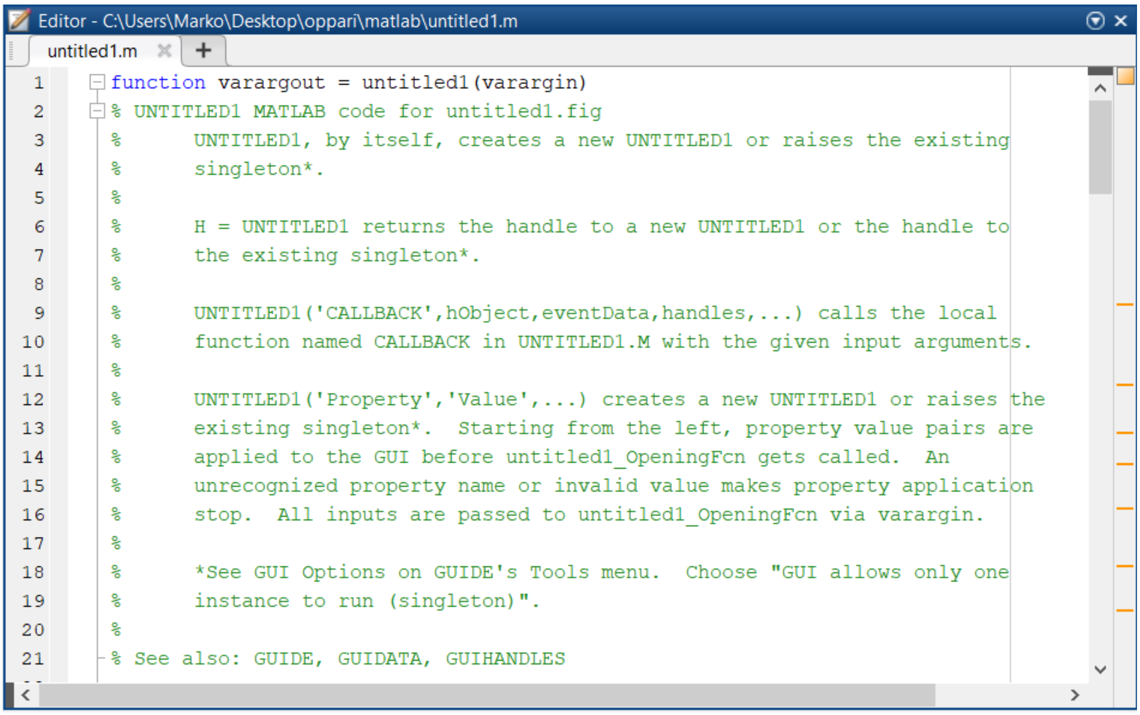


Image 11. F-file (figure). In the left side of the editor is possible to choose components for the layout.

Image 11 shows a F-file with components of edit texts, axes, static text and push button. By right clicking the component with mouse, it is possible to inspect the properties and make changes for the specs to change font, values and other information. In properties, the important is *tag*, which is used in the program for indicating the component. To enter to the editor and start programming, it is needed to right click and make a *Callback* for a component or the window. After saving a F-file it automatically creates the M-file and it is possible to start coding it. The M-file contains the MATLAB functions that control the applications behaviour. (MathWorks: Create a Simple App GUI using Guide 2019)



```
function varargout = untitled1(varargin)
% UNTITLED1 MATLAB code for untitled1.fig
%      UNTITLED1, by itself, creates a new UNTITLED1 or raises the existing
%      singleton*.
%
%      H = UNTITLED1 returns the handle to a new UNTITLED1 or the handle to
%      the existing singleton*.
%
%      UNTITLED1('CALLBACK',hObject,eventData,handles,...) calls the local
%      function named CALLBACK in UNTITLED1.M with the given input arguments.
%
%      UNTITLED1('Property','Value',...) creates a new UNTITLED1 or raises the
%      existing singleton*.  Starting from the left, property value pairs are
%      applied to the GUI before untitled1_OpeningFcn gets called.  An
%      unrecognized property name or invalid value makes property application
%      stop.  All inputs are passed to untitled1_OpeningFcn via varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES
```

Image 12. The program made from layout by making a *Callback* from F-file.

Interaction between menus, buttons and sliders happens from the M-file from the MATLAB-code written in the code editor, and the F-file gives easy to handle figure for the developer. After the program is ready, it is possible to execute it from *Run* button. Later, it is always possible to make new changes for the layout and add new components.

Difference to the programmatical way is an own file for figure and small differences in the functions. Both programs can show up the MATLAB diagrams, including necessary menus, buttons and sliders to handle the data behind the User Interface. More accurate information about building applications with GUIDE are possible to find in the website of mathworks.com.

3.2.3 App Designer

App Designer is the newest way in MATLAB to build applications, which allows a user to make applications which can be compiled to Web Applications. It is released in the version R2016a and have a modern outlook comparing to GUIDE or programmatically way. The buttons and other components have a modern outlook and building applications itself have been made easier to approach. It is possible to do applications with less knowledge of programming. App Designer is an external environment, and to enter to the working environment it is needed to write in the command window *appdesigner*. (MathWorks: Ways to build Apps 2019)

In App Designer the file mode is different than in other versions. As in programmatical way it is **.m** and in GUIDE it is with **.m** and **.fig** file, in App Designer the file mode is **.mlapp**. The working environment is an external and the code of the application is written in external editor that is integrated with a layout view. Layout window is called *Design View* and the editor is called *Code View*. In making of a layout for the application, the visual components are added with drag and drop method. The application gives hints in alignment to get precise layout. (Nouman Azam 2016)

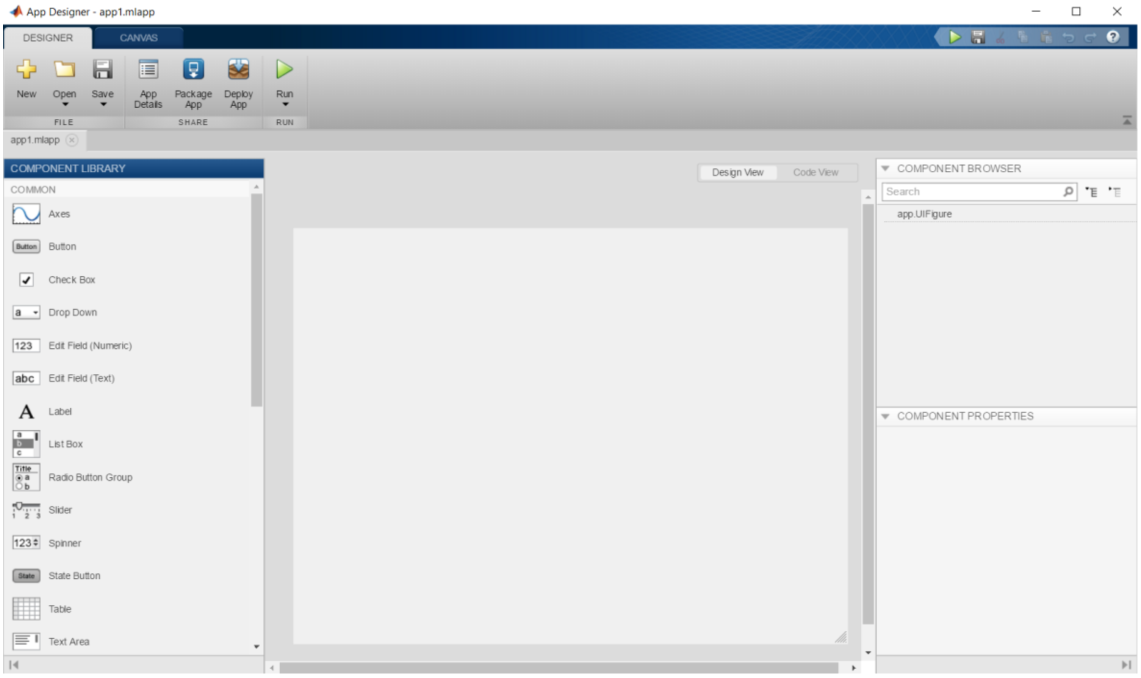


Image 13. MATLAB App Designer environment.

The design of the components has a modern look and the amount of the components are wider. App Designer includes new components such as gauges, lamps, knobs, switches and indicators. To handle these components, the software generates automatically object-oriented code. Functionalities of the program are handled with a *callback* function. *Callback* is a function, which executes when a user interacts with a user interface component. *Callback* is possible to make for all the components, except the ones which only displays information. (MathWorks: Write Callbacks in App Designer 2019)

3.2.4 Conclusion

When building an application with MATLAB, in the first is needed to choose the way how to build the application. The chosen way to build an application may later affect the functionalities and make the limits in working. For example, all the components aren’t available in all the methods. Also, the choosing of the method defines the working methods. Even the process in making an application with MATLAB is quite similar in every method, all the methods of building applications have their small differences in working and programming. In programmatically can be easier to maintain the project and it limits the least comparing to the two other methods. Negative in making programmatically is the missing visual support, which exist in GUIDE and App Designer.

The biggest difference between the method’s is that the applications which are made with the App Designer, are possible to compile for web applications. The applications made with programmatically and GUIDE, are possible to compile only for individual applications, .exe.

As the original programs of the project are in .m file, I found in App Designer problematic to make the GUI of the project with it. The first idea was to build the GUI in programmatic way, because of the same file mode and clear working method to focus on the code. But after having some problems with variables in a programmatic way and managed to make them easily with GUIDE, it was selected to the working method for the project. GUIDE also offered visual support, flexible layout planning and easily understandable functions and tags.

3.3 Designing User Interface and Experience

Important part of building Graphical User Interface is to specify how is the interface designed. Inside the software industry is common that the design of the user interface is made by software engineer, who have little or not at all knowledge about the UI design and no guidance or support from UI designers.

In graphical designing, people talk about UI and UX design. UI means User Interface, which means how is the surface, how it looks and feels visually. UI is about branding, which includes colours, schemas, fonts, logos and style. User Interface is how the product look, for example, controls, navigation, interaction, graphic design and layout, responsiveness and management. User Interface is everything which is seen with the eye. (Jeff Johnson 2007, Introduction)

UX, which means User Experience, is how it works and feels practically. It means the experience what the using of the product or service gives. It is about feeling, how the using of the application feels, is it easy-to-use, are there problems in using and about general satisfaction. In the last years, user experience has defined the success for the products and services. As an example, Apple's tactic in markets is much based on User Experience. (Joonas Virtanen 2016)

The world is full of products with a bad user experience design. One common example about the bad design are doors. In a well-designed door the user does not make error in opening of the door and do not try to push it when it is needed to pull. Basically, in the digital side the rules are same, the applications or web browser's elements needs to be logical and clear for good user experience. The hierarchy needs to be correct and the user should not need to use onboards in the using even though it is the first use, or the service is complex. The clearness of the user experience can be helped with the visual colours, tips or with an information. (Joonas Virtanen 2016)

The common question in UX design is *"what the usability is compared to desirability"*. Usability is the base level of the user experience but without desirability, it can give unlikely user experience. In 2008 at the Nielsen Norman Group Conference in Amsterdam is defined the four simple levels of User Experience, which are:

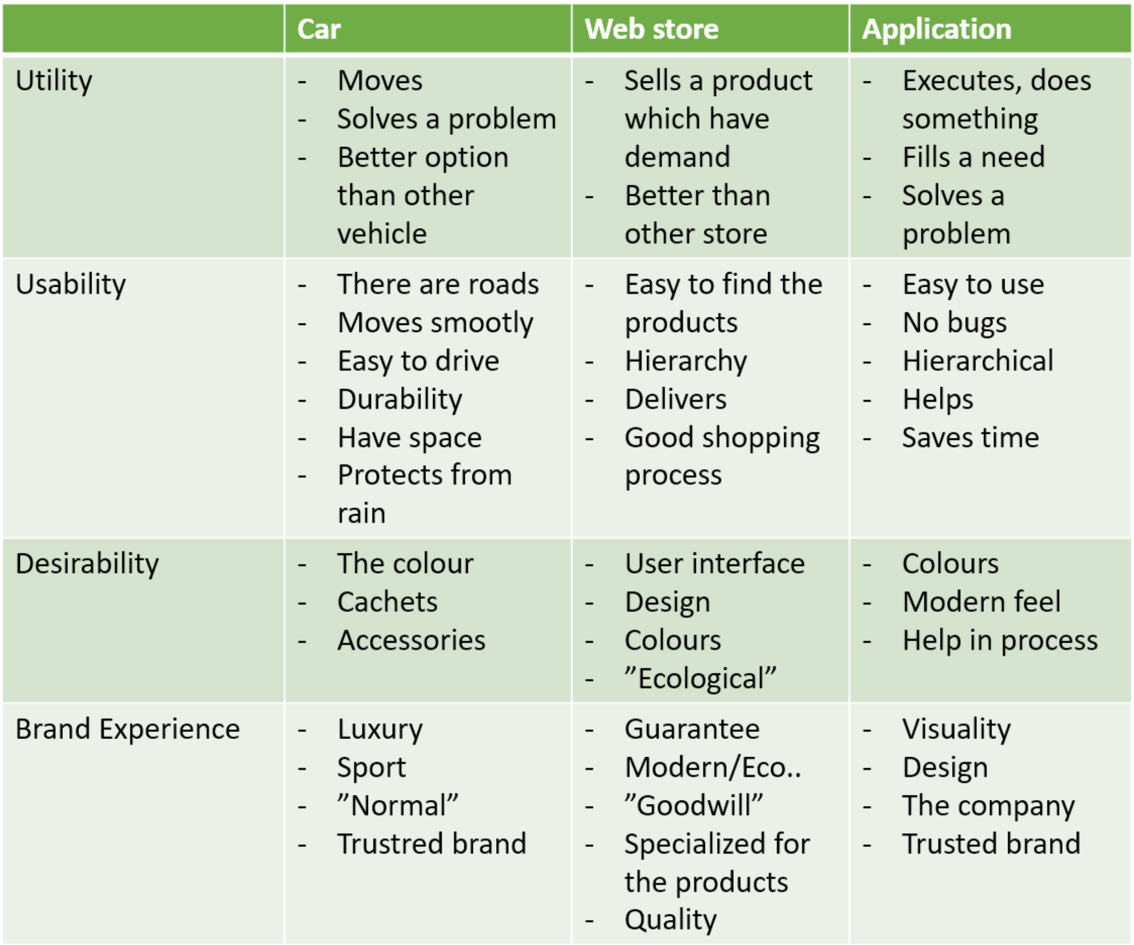
- Utility
- Usability
- Desirability
- Brand Experience

As the first thing in user experience is utility, which like to ask, *“are there a use for a product”.* Utility of a product or service means it provides value for a user. For a car, the utility means that the car can move. Basically, when it moves, it is useful for user and it is a first wanted feature from a car. Also, if someone wants to buy a car, the buyer has a need for it, or it solves a problem. If it solves a problem for something, there is utility in the product. It gives the value for the product or service. The Utility of a product can be identified as, does it get the job done, do someone have need for it or do someone want to use it. One important thing about utility also is, does it give more value than some other vehicle. For example, a bicycle or a horse. (Interaction Design Foundation)

After there is utility for a product or service, there must be usability. For the user, the service or the product needs to be easy and intuitive to use. For example, in a web store the usability’s are the experience in shopping cart and the paying. For a car the usability means that there are roads for it. For a car it can also mean that the car moves smoothly, and it have a good touch when driving. The usability is about giving the satisfaction for the user and about filling the need without having a problem in using. (Interaction Design Foundation)

As the common question is how the usability is compared to desirability, and which of those two matters more in the end for a user when making a choice in selecting between two products or services. In most fields of business, there are competitors. If comparing a different car companies, electronic devices, travelling services or breads, there are different brands or companies which offers exactly a similar product or a service. In general, the product of different brands usually offers the same utility and usability than the others. Then in the selecting, the desirability makes a place. Desirability is related to emotional appeal. Desirability is usually about the visual design, colour, shapes or about the reputation of being normal, luxury, ecological or something else. One way to improve desirability is marketing, which can make the user feel desire for a product or a service. (Frank Guo 2012)

The last level of these four is brand experience, which goes much hand-in-hand with desirability. Brand experience really much answers for the question, *"Does the user feel good about the product and the company/brand that makes it?"*. The experience can be the quality of the product, quality of the service or the guarantee that the company provides. Other effecting thing for the brand experience can be the ecological or luxury reputation. (Interaction Design Foundation)



	Car	Web store	Application
Utility	- Moves - Solves a problem - Better option than other vehicle	- Sells a product which have demand - Better than other store	- Executes, does something - Fills a need - Solves a problem
Usability	- There are roads - Moves smootly - Easy to drive - Durability - Have space - Protects from rain	- Easy to find the products - Hierarchy - Delivers - Good shopping process	- Easy to use - No bugs - Hierarchical - Helps - Saves time
Desirability	- The colour - Cachets - Accessories	- User interface - Design - Colours - ”Ecological”	- Colours - Modern feel - Help in process
Brand Experience	- Luxury - Sport - ”Normal” - Trustred brand	- Guarantee - Modern/Eco.. - ”Goodwill” - Specialized for the products - Quality	- Visuality - Design - The company - Trusted brand

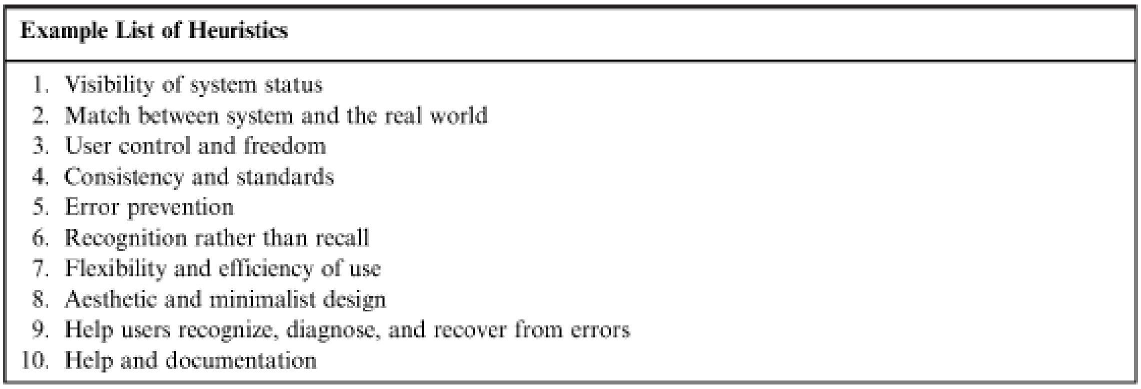
Image 14. Utility, Usability, Desirability and Brand Experience of different products.

Image 14 explains the four different levels with the different type of products. These four different levels can be used basically for every product or service.

3.4 Heuristic Evaluation method

This chapter is based on the “Chapter 1. Heuristic Evaluation” of the book “User Interface Inspection Methods”, which is made by Chauncey Wilson and Published by Morgan Kaufmann in 2013.

The meaning of the word *“heuristic”* comes from Greek language word *“heureka”*, which means “find” or “discover”. Heuristic Evaluation method focus on discovering the product and find out common usability problems and recognize awareness of them. Heuristic Evaluation consist from three parts; planning, conducting and after. Heuristic Evaluation is a popular method to inspect applications usability and find usability problems in the user interface design. The method evaluates a user interface with a set of principles and common-sense rules. The method evaluates a product, a prototype or specifications against a brief list of “questions”. Heuristic evaluation is possibly to do as an individual or with a team. (Chauncey Wilson 2013, Chapter 1)



Example List of Heuristics

1. Visibility of system status
2. Match between system and the real world
3. User control and freedom
4. Consistency and standards
5. Error prevention
6. Recognition rather than recall
7. Flexibility and efficiency of use
8. Aesthetic and minimalist design
9. Help users recognize, diagnose, and recover from errors
10. Help and documentation

Image 15. Example list of Heuristics. A Set of Heuristics from Nielsen (1994a)

There are two main reasons to do heuristic evaluation. The primary goal is to avoid usability and design problems, especially when it is still easy to do with a low cost. The second is to train members of the team to recognize potential problems in design and detects them in early of the process. In these situations, heuristic evaluation fits well:

- Limited or no access to users.

- When having an easy access to potential evaluators of the design.

- Need for fast review.

- Even using other testing, the heuristic evaluation gives other perspective for the testing and can find more problems or noticeable things.

- There is a lack of funding in usability testing.

Strengths	- Simple to explain - Fast and cheap - Similar to software code inspection - Does not require special resources - Increase awareness of common usability problems
Weaknesses	- Different evaluators often find different problems, makes possibility to ”false positives” - Finds only usability problems (are they real problems always?) - May not scale with complex interfaces - Evaluators may not be the actual users - Depended about the quality and experience of the evaluators - Doe not offer a solution

Image 16. Strengths and weaknesses of the Heuristic Evaluation. (source: Chauncey Wilson 2013, Chapter 1)

3.4.1 Planning

After choosing to perform Heuristic Evaluation, it begins from planning. As Heuristic Evaluation is considered as informal with an advantage that it “does not require advance planning”, but especially when conducting first times it is required. The planning can be followed from 7 basic steps. (Chauncey Wilson 2013, Chapter 1)

1. **Choose a team of evaluators.** The best evaluators are the people with experience of the similar product or the users of the product. Professionals with experience can see easily the errors and mistakes from the UI/UX. Evaluating should not be depended from a single evaluator. To get reasonable data from evaluation, the size of members would have to be minimum of three to five people. Wide and good knowledge of the evaluators makes better results. Also, when selecting evaluators, it is important to avoid “familiarity blindness” by choosing into the evaluation team some fresh eyes.

2. **Decide which Heuristics are most useful for your evaluation.** There are different sets of lists of heuristics. In the Image 15. is shown the Nielsen’s original set of heuristic but there are more available. The choosing can depend from the subject and the goal of the evaluation. In choosing a set of heuristic, it is important to consider the relevance of a list for the product, understandability of the members, memory aids to aim for the goal, the validity and would there be actual impact on performance in the end.

3. **Develop an infrastructure for collecting, organizing, tracking, and reporting on the results of the heuristic evaluation**. Better environment makes better results. Heuristic evaluation infrastructure consists from forms to collect problems (paper or online), brief description, from the good description of the product or problems and the possibility problems.

4. **Conduct a short training session with potential evaluators who have not been part of earlier evaluations**. This is an optional step but it is an activity which increases the quality of the reviews. Training session means introducing the process of the heuristic evaluation, explaining the used heuristic, showing the forms and the level of expected detail.

5. **Provide the evaluators with some context to help them understand the business objectives, the personas or user profiles, the user goals and tasks, and the environments where the product will be used**. Meaning to consider the relevance background for the evaluators. The amount of the background information depended of the complexity of product.

6. **Provide the team with an overview of the prototype or working system that they will be evaluating**. This includes the overviewing of the use of the product, major features, limitations which are needed to know, the realistic level of the product or data (if not the end-use of product or data), where to ask support and how to get access into the system.

7. **Choose which approach to your heuristic evaluation is most appropriate given the state of the product**. The basic approaches are to create an important task scenario, giving a set of goals and asking from the evaluators a specific thing (for example, UI objects, pages in website, windows, menus, …). Most powerful is to ask evaluators to evaluate a product based on all the three mentioned approach and report the problems which they encounter. Obviously, it takes more planning and effort. (Chauncey Wilson 2013, Chapter 1)

3.4.2 Conducting

Conducting of the heuristic evaluation means carrying out the practical part of the heuristic evaluation with a team of elevators. The value of conducting a heuristic evaluation depends strongly of the experience and quality of the evaluators. The team who conducts a heuristic evaluation should have relevant knowledge of the domain and users, and in a best case to be the actual users of it. The evaluation can be conducted after problem definition and giving the best results with good planning (Chapter 3.4.1). It is also important to define the goals and the scope after conducting. Basically, the scope of the heuristic evaluation is to find problems in usability and improve the user experience. (Chauncey Wilson 2013, Chapter 1)

In a book User Interface Inspection Methods are shown these seven steps to be followed in conducting of a heuristic evaluation:

1. **Orient the evaluators.** Meaning sharing the orient information and providing the UI materials for the evaluation (screenshots, a paper prototype, wireframes, product or even a competitive product). Answering for the questions that evaluation team might have and showing the schedule of the evaluation.

2. **Ask the evaluators to conduct individual evaluations of the UI.** As the evaluations are generally done individually, there is probability that some of the peoples does it “wrongly”. It can be suggested the evaluators to:

   a. Familiarize with background information (goals, personas, environments descriptions).

   b. Walk through the task scenarios.

   c. Review additional parts of the product which are not part of the task scenarios.

   d. Suggests the evaluators in a problem situation to contact the leader of the evaluation.

3. **Advise your evaluators to list problems separately.** The problems can be global or local which is important to know. Also, if the same problem exists in many places or a problem exist because the product is still on prototype mode, evaluators need to be notified.

4. **If your heuristic evaluation will last over several days, consider reconvening your evaluation team at the end of the first day of review so that issues about the procedure or product can be discussed and ironed out before they have gone too far in the evaluation.**

5. **Collect the heuristic evaluations form containing the list of problems found by each evaluator.** Tell the evaluators that you may ask them later about their answers. (Chauncey Wilson 2013, Chapter 1)

3.4.3 After

When planning is far behind and evaluation conducted successfully, it is time to get benefits from The Heuristic Evaluation. Probably the team and the leader have already noticed problems and uses them for making improvements. To get all advantage from Heuristic Evaluation, there is five step list what to do after conducting heuristic evaluation. (Chauncey Wilson 2013, Chapter 1)

1. **Compile the individual lists of problems into a single list, and then decide on how to arrive the severity ratings.** If forms hold rating scales, it is better to calculate them for an average of the scores and compare the results. Good solution is to set up meeting with evaluators and ask from each why they responded as they responded. It gives opportunity to understand if there is a lot of variability in the answers.

2. **Organize the problems in a way that is useful to the product team.** It is good to list and highlight the more serious problems and then organize the individual problems. When conducted, the summary of recommendations needs to be shared for product team depending on their role. Design problems for designers and technical problems for technical team.

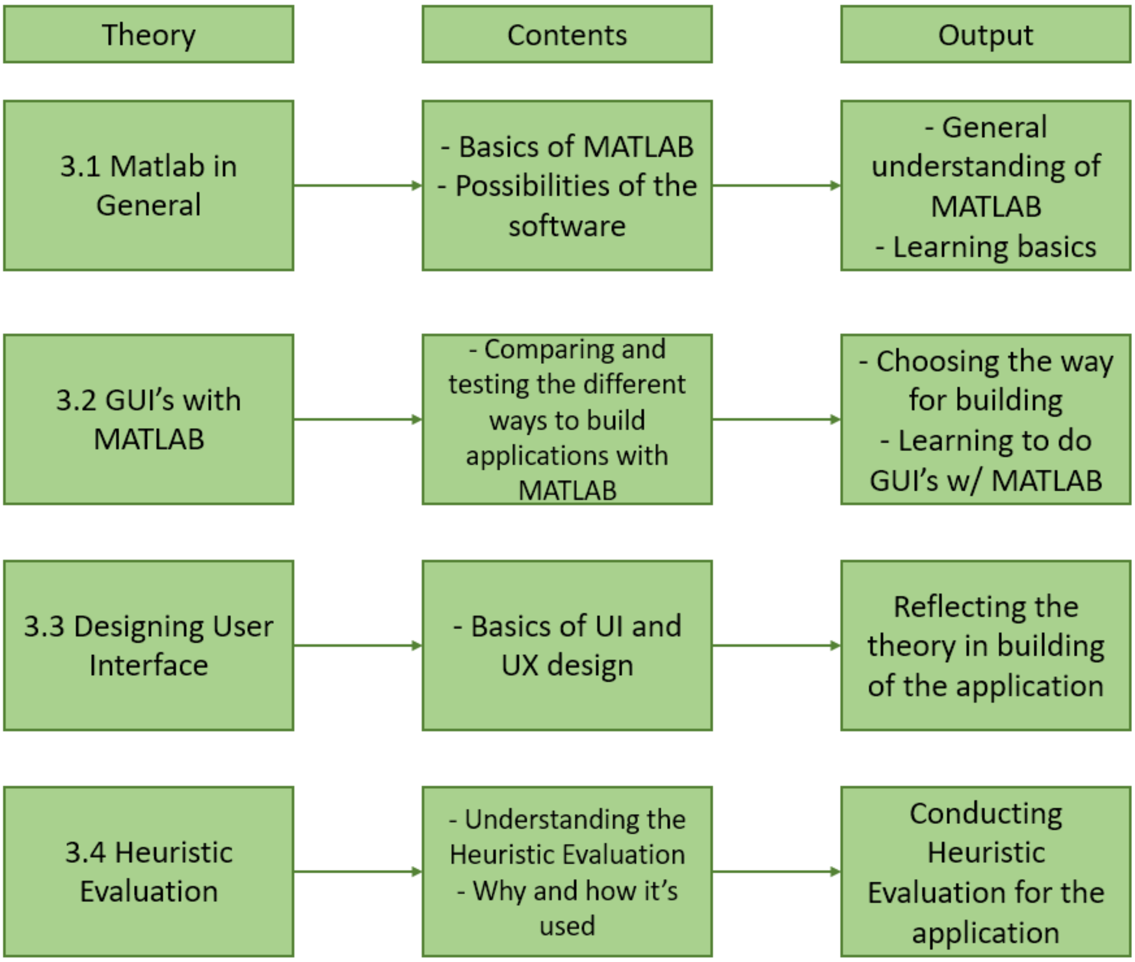
3. **Consider whether you want to have a group meeting of the evaluators, developers, and designers to prioritize the results and discuss recommended solutions.**

4. **Catalog, prioritize, and assign problems, themes and issues to the appropriate members of the product team.** It is important to arrange meetings to review potential solutions especially for the more serious problems and take care of the individual problems also.

5. **Validate the changes to the product with user tests, beta feedback, or other evaluation methods whenever possible.** In this kind of improvements, it is common that the benefits are not considered totally. Take care that in the end the evaluation is carried out successfully. (Chauncey Wilson 2013, Chapter 1)

3.5 Conclusion of the theory

The theory of the thesis consists from 4 parts, getting to know the MATLAB in general, building of graphical user interfaces with MATLAB, designing of the user interface and experience and from The Heuristic Evaluation method. The use of the theory is explained in the image below, by showing the subject of the theory, the contents and the output.



Theory	Contents	Output
3.1 Matlab in General	- Basics of MATLAB - Possibilities of the software	- General understanding of MATLAB - Learning basics
3.2 GUI's with MATLAB	- Comparing and testing the different ways to build applications with MATLAB	- Choosing the way for building - Learning to do GUI's w/ MATLAB
3.3 Designing User Interface	- Basics of UI and UX design	Reflecting the theory in building of the application
3.4 Heuristic Evaluation	- Understanding the Heuristic Evaluation - Why and how it's used	Conducting Heuristic Evaluation for the application

Image 17. The use of the theory.

4 Project: Creating the Application

The graphical user interface which is built in this chapter is deployed to the application. Because of that, sometimes the name changes between application and GUI even though they mean the same product. The scope of the GUI is to unite two already existing programs which makes reports of the air quality. From the GUI will be possible to choose the needed variables for executing the original programs.

Creating of a graphical user interface includes many steps which need to be considered. In this project the focus has been first in understanding of the problem and familiarizing with necessary tools. After having a general knowledge, the project proceeded for creating of the application itself.

As the graphical user interface is going to be made with MATLAB, it is important to understand the MATLAB computer software and how to build graphical user interfaces with it. As MATLAB have three different method to build GUI’s, the selected method for this project is creating it with Graphical User Interface Development Environment, called GUIDE. The main reasons for choosing GUIDE are easier workflow comparing to the programmatical way and giving more flexibility than App Designer

This chapter focuses on building the application. It begins by showing the process of creating before it moves forward to the designing & planning and later to the ready-made standalone application. After the application is ready, it is tested with the method called Heuristic Evaluation to inspect potential usability problems.

4.1 Steps in the creating of the application

The workflow of creating the application is shown in this chapter. The process map gives a picture of the whole process of building the application from practicing to the usability inspection of the ready application.

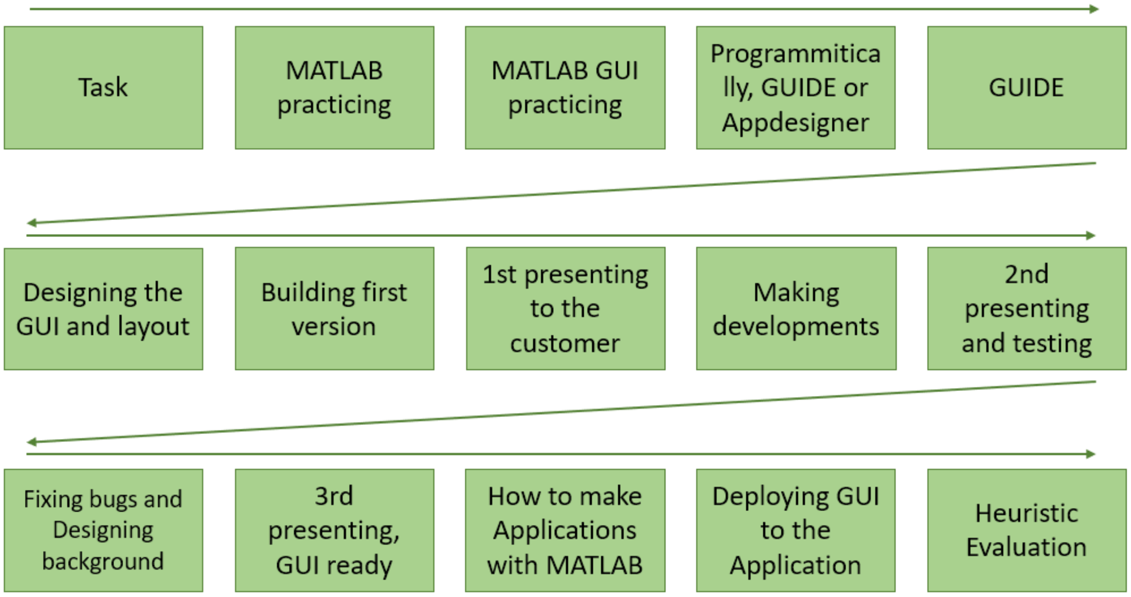

Task
MATLAB practicing
MATLAB GUI practicing
Programmitically, GUIDE or Appdesigner
GUIDE
Designing the GUI and layout
Building first version
1st presenting to the customer
Making developments
2nd presenting and testing
Fixing bugs and Designing background
3rd presenting, GUI ready
How to make Applications with MATLAB
Deploying GUI to the Application
Heuristic Evaluation

Image 18. Creating of the application in process steps.

After the task was clear, the project started with practicing the using of MATLAB software. In the beginning, there was not any experience of using MATLAB, so the project started by familiarizing with it. Familiarizing with MATLAB was done from the MathWorks.com web page, which is the homepage of the software. Graphical User Interface building is learned from the series of videos in “MATLAB App Designing: The ultimate Guide for MATLAB Apps” made by Nouman Azam.

After familiarizing and founding the best method to make application of the project, the project and product started to run out by own by solving new problems of the building of the application and settling new meeting schedules with the customer. In the simplified Gantt chart (Image 7.) is shown all the main steps of the project.

4.2 Designing & Planning

After the goals and requirements are defined the project focused to take them into account in planning and designing. Basically, the GUI need to reach the defined goals and be built with MATLAB to get easier access to the original program. As first in researching how to build applications with MATLAB (as there is 3 different ways) the best solution founded from GUIDE tool, because of the easier workflow comparing to the programmatical way and giving more flexibility than App Designer.

In planning need to consider the functionalities, application workflow and layout & design. The application functionalities are chosen from 3 variables, which are the report model (daily or monthly report), the date and the measurement. The report model consists from variables monthly and daily and being the “first” variable of the program, meaning, before choosing that variable, other variables in layout are invisible. The point of this is to avoid making errors in choosing. Second variable date, is chosen from Pop-up menu components, depending from the first variable, an exact day or a month. Third variable measurement, meaning the quality of air giving in particles PM2.5 and PM10 are chosen from Radio buttons. It is possible to choose only one of them or both.

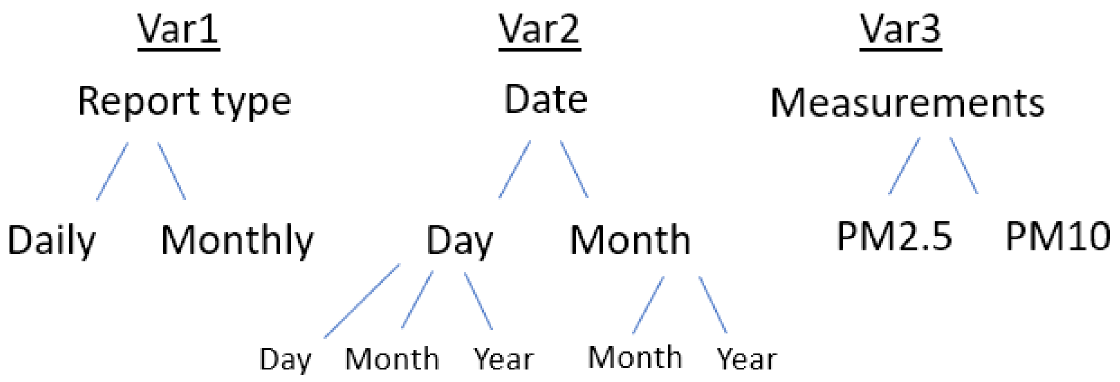


Image 19. Variables to be selected from the GUI.

The components and variables need to communicate with themselves to work properly and to offer good user experience. The first important part is the main window, which includes the components to choose variables and a component to push Run the program. The Application need to communicate with a user to give a better experience. After clicking Run, the application would ask about Confirmation or Alert about the existing Error. If everything is correct, clicking *Yes* continues the program and *No* would return to main page. Because the running of the program will take several minutes, it is useful that the user knows in which stage the program goes. And when the program is ready, it will notify if everything was executed correctly or not.

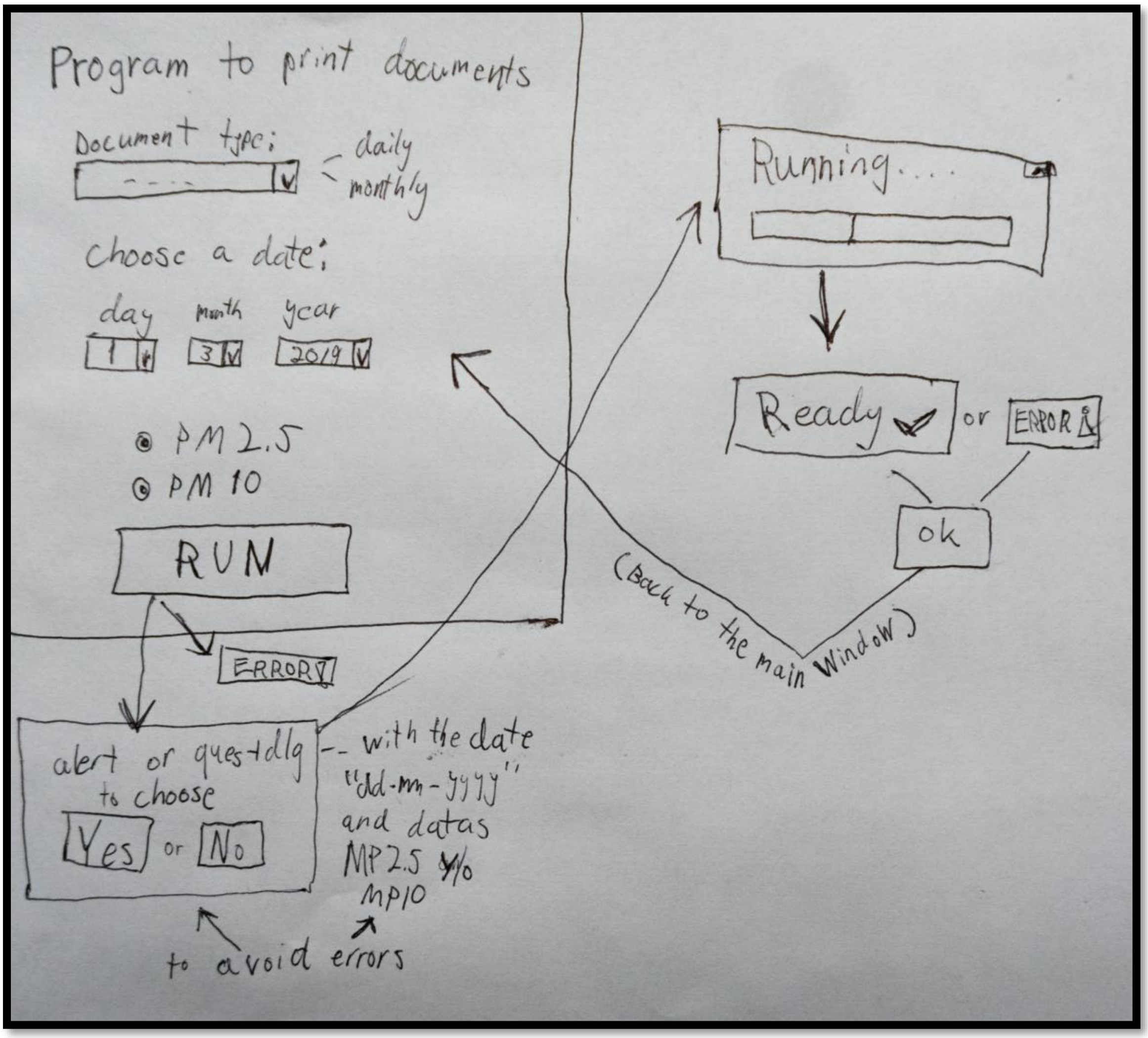


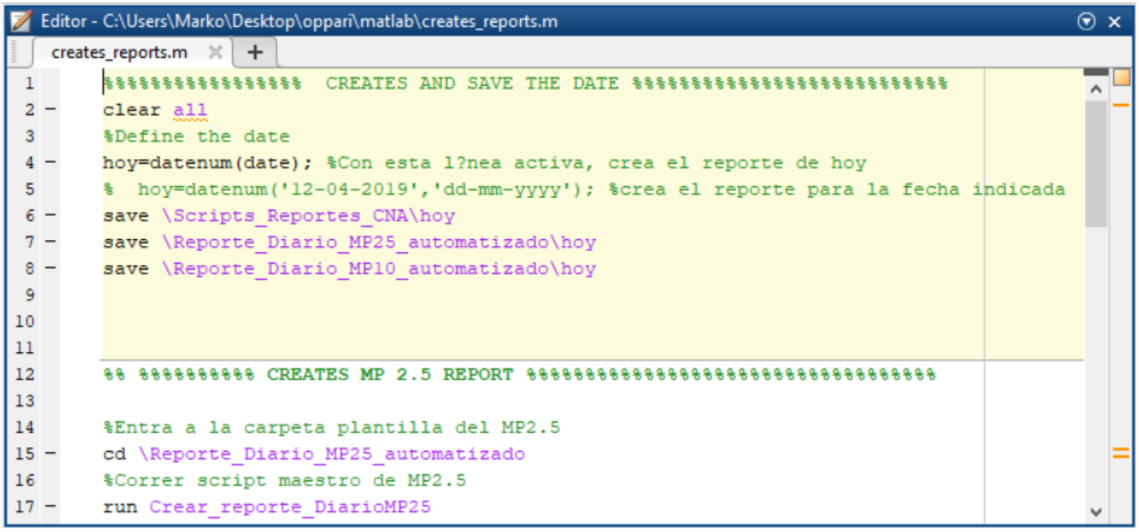
Image 20. Wireframe of the application.

Layout & design of the application is much depended about the functionalities and the workflow of the application. Component types are focused on variable type considering the variable value. Order of components follows the rational order, focusing first in project model (daily or monthly), then the date (month or exact date) and after that, the measurements.

In layout design the hands are free with the sizes of components and fonts without requirements from the customer. As one of the requirements is a good workflow, the font's followed the basic of fonts (MS Sans Serif) of this kind of application and the layout with the program avoids making errors. In design of the application, the application is required to have “the theme of The Ministry of The Environment”, using similar colours including their logo. They are applied later in building.

4.3 Creating of the GUI

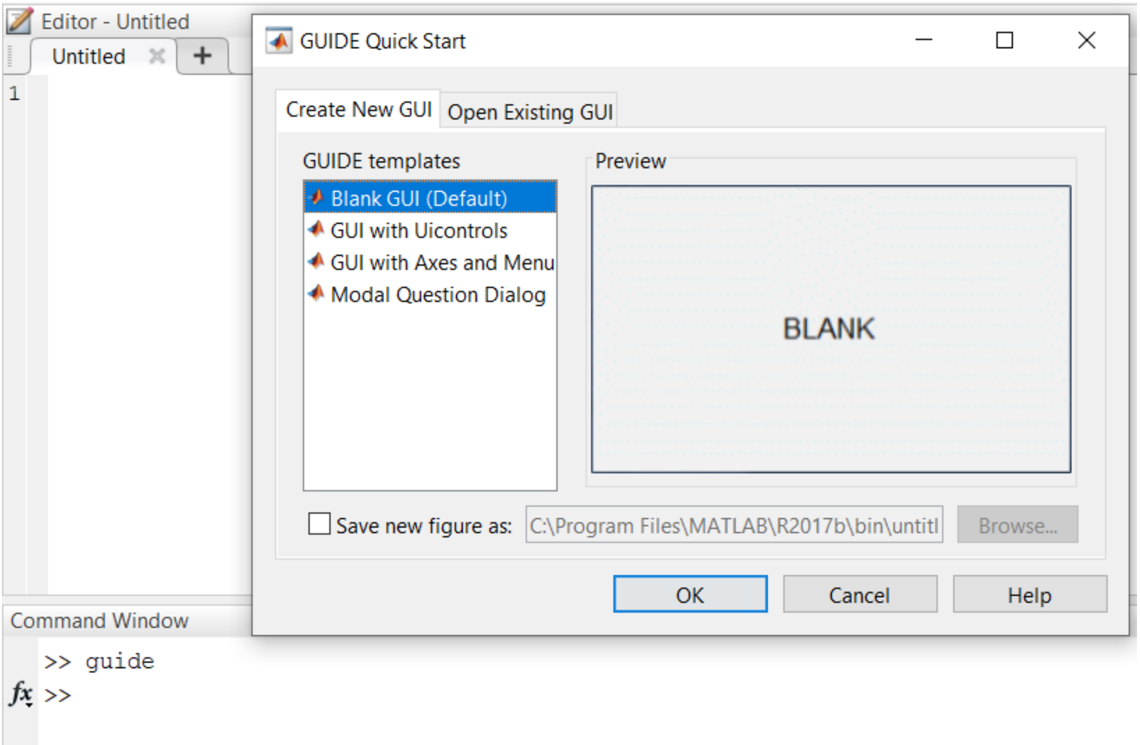
The creating of the graphical user interface is focused to the original code and making the graphical user interface over it. Originally, there are 2 different programs which are for Daily reports and Monthly reports. After opening one of these programs, it is needed to make change in the variable *date* inside the program, depending if daily to choose the exact day or if monthly, to choose a month (1st day of the next month is the earlier month). After that, the program always run’s the documents of reports PM2.5 and PM10. With a graphical user interface, the goal is to make all these three different variables easy to select.


Editor - C:\Users\Marko\Desktop\oppari\matlab\creates_reports.m
creates_reports.m
+

```
 1      %%%%%%%%%%%%%%%%%  CREATES AND SAVE THE DATE %%%%%%%%%%%%%%%%%%%%%%%%%%%
 2 -    clear all
 3      %Define the date
 4 -    hoy=datenum(date); %Con esta l?nea activa, crea el reporte de hoy
 5      %  hoy=datenum('12-04-2019','dd-mm-yyyy'); %crea el reporte para la fecha indicada
 6 -    save \Scripts_Reportes_CNA\hoy
 7 -    save \Reporte_Diario_MP25_automatizado\hoy
 8 -    save \Reporte_Diario_MP10_automatizado\hoy
 9
10
11
12      %% %%%%%%%%%% CREATES MP 2.5 REPORT %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
13
14      %Entra a la carpeta plantilla del MP2.5
15 -    cd \Reporte_Diario_MP25_automatizado
16      %Correr script maestro de MP2.5
17 -    run Crear_reporte_DiarioMP25
```

Image 21. The original program for daily reports. In line 4, is chosen the current day report. The program runs the reports of PM2.5 and PM10.

When making applications with MATLAB GUIDE, it consists from 2 files, .m file for the code and .fig file for the figure. Figure is the appearance of the application with all the components and layout of them. M-file involves the program, which manage the components in the F-file (figure) with MATLAB functions. Making the application with GUIDE begins with writing *guide* to console and choosing template, in this case the Blank template.



□ Save new figure as: C:\Program Files\MATLAB\R2017b\bin\untitl

```
>> guide
>>
```

Image 22. Starting the building of GUI with GUIDE in MATLAB.

Subsection 4.2 designing and planning shows the wireframe of the application. Because the guide tool is effective in planning and making the wireframe itself, there was not need for individual digital wireframe. The first step in building graphical user interface with GUIDE, is to make the figure with GUIDE tool. Image 23 shows the main window of the application in F-file. As with GUIDE tool the application consists of two files, F-file and M-file, the M-file composes automatically after saving the F-file. Also, when trying to make functions to the user interface components in figure by making *callback function*, the program obliges to save and makes automatically the M-file.

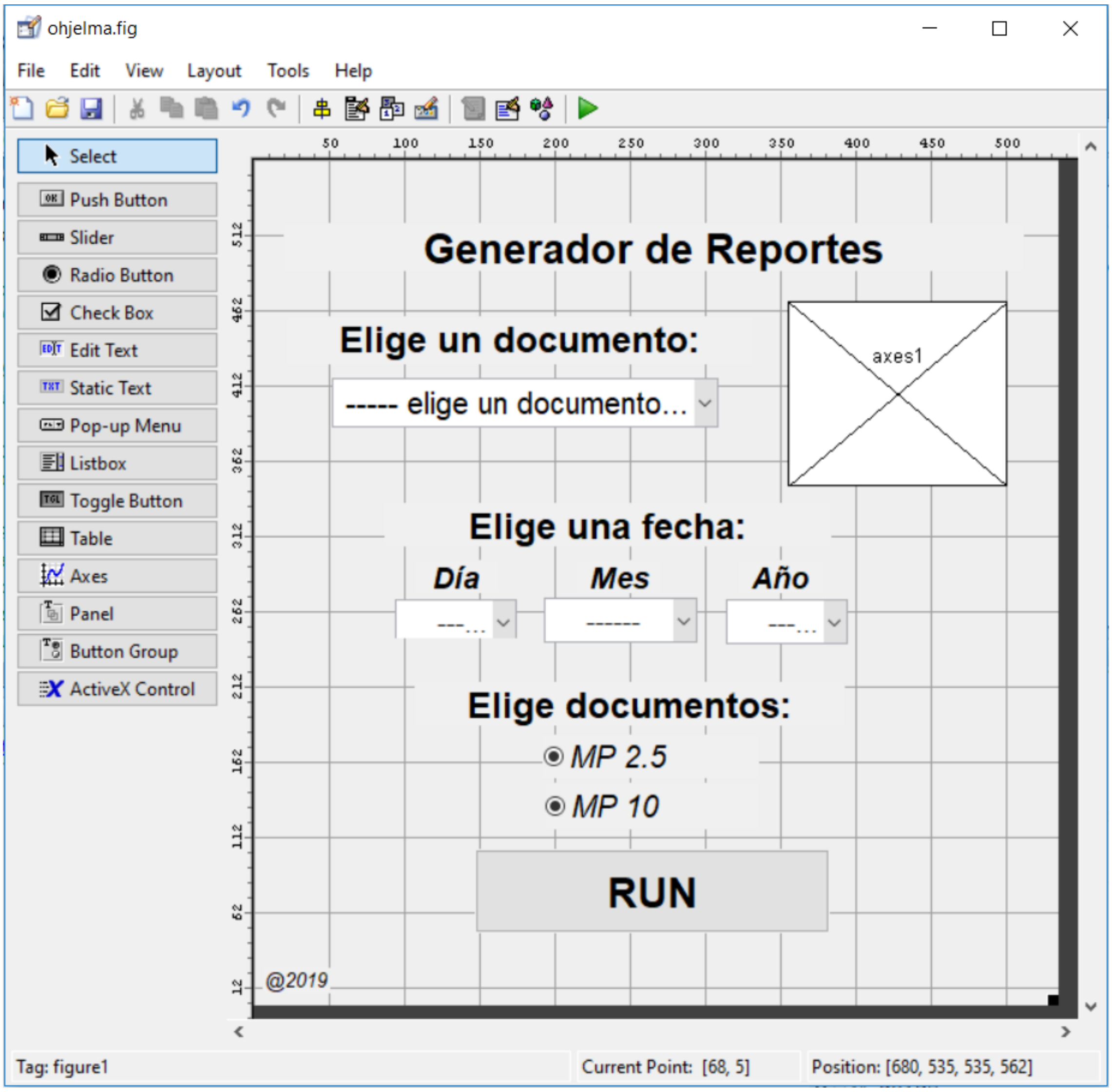
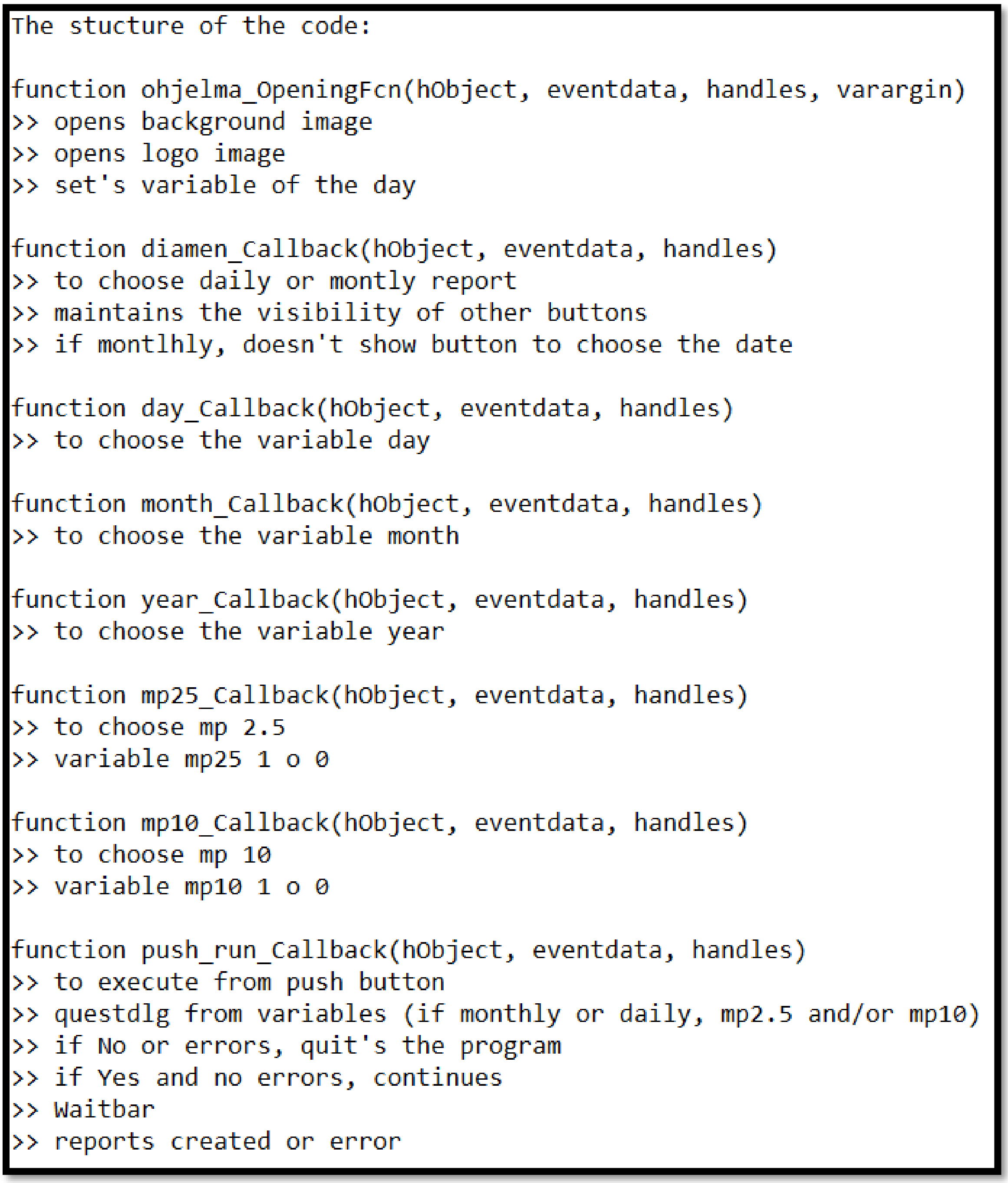


Image 23. Creating GUI with GUIDE begins with making the figure. In the figure are added the needed user interface components. Making functionalities happens by saving or making callback function to F-file, and then coding with MATLAB functions to the M-file. The specs of the different components (tag, value, fonts, etc..) can be changed by right-clicking and selecting “Property Inspector”. In this GUI the language is Spanish.

In M-file, the code consists from the function of the program, Opening function and functions of the components. Opening function is the first function in the program and the role of it is to perform the code inside the opening function before the User Interface is shown for a user. In the graphical user interface of the project, it is used for visualizing the program, by setting a background image for the whole application and to input customer logo for *axes1*, which is seen in the Image 23. above. The other functions in the program are *callback* functions to the existing buttons.

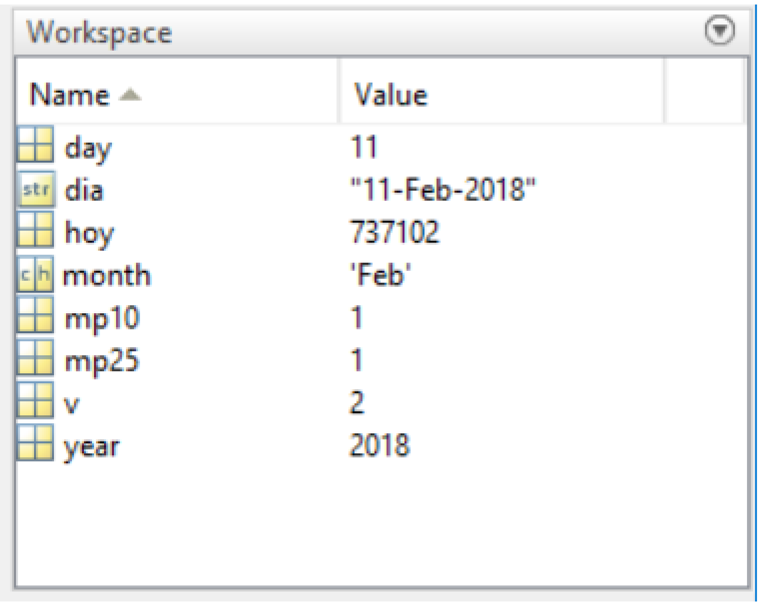
The actions in the program are performed from *callback* functions of the buttons. As the code of the program would be too long to show, in the Image 24. is shown the structure of the program and explained the function of them. Basically, every component in the program have the function except the Static Texts.



```
The stucture of the code:

function ohjelma_OpeningFcn(hObject, eventdata, handles, varargin)
>> opens background image
>> opens logo image
>> set's variable of the day

function diamen_Callback(hObject, eventdata, handles)
>> to choose daily or montly report
>> maintains the visibility of other buttons
>> if montlhly, doesn't show button to choose the date

function day_Callback(hObject, eventdata, handles)
>> to choose the variable day

function month_Callback(hObject, eventdata, handles)
>> to choose the variable month

function year_Callback(hObject, eventdata, handles)
>> to choose the variable year

function mp25_Callback(hObject, eventdata, handles)
>> to choose mp 2.5
>> variable mp25 1 o 0

function mp10_Callback(hObject, eventdata, handles)
>> to choose mp 10
>> variable mp10 1 o 0

function push_run_Callback(hObject, eventdata, handles)
>> to execute from push button
>> questdlg from variables (if monthly or daily, mp2.5 and/or mp10)
>> if No or errors, quit's the program
>> if Yes and no errors, continues
>> Waitbar
>> reports created or error
```

Image 24. Structure of the code in the MATLAB M-file.

Around the functions, all the variables are global variables. Having global variables can be problematic, when there are lot of variables and the variable can be forgotten (or the reason of it). In this user interface, it is founded as solution to maintain all the components and to be sure, that the information's of the components are correct inside the MATLAB workspace before clicking *RUN*.



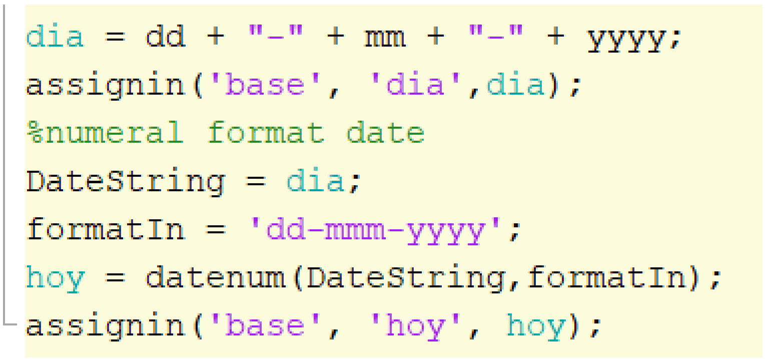
Workspace

Name	Value
day	11
dia	"11-Feb-2018"
hoy	737102
month	'Feb'
mp10	1
mp25	1
v	2
year	2018

Image 25. Variables of the program in MATLAB workspace.

Image 25. shows the variables of the program. With these variables can be formed all the different choices of the needed variations to execute the desired documents from the data.

Basically, the variables consist from three main variables which are *v, hoy* and *mp25/mp10*. Variable v, is meant in the beginning for choosing the daily or monthly report, being 1 when not selected, 2 when daily and 3 when monthly. When not being selected, the rest of the components are not visible. When daily, all the components are visible and when monthly, except the component to pick the *day* all are visible. In monthly report, the variable *day* is automatically with value 1 and it is not possible to choose the day from the component (component invisible). The variable dia (date), is the variable which consist from selected day, month and year. The variable *hoy* (today), comes from the value of the day which is formatted for serial number of the day. The variable *hoy* is used in executing/running the program.



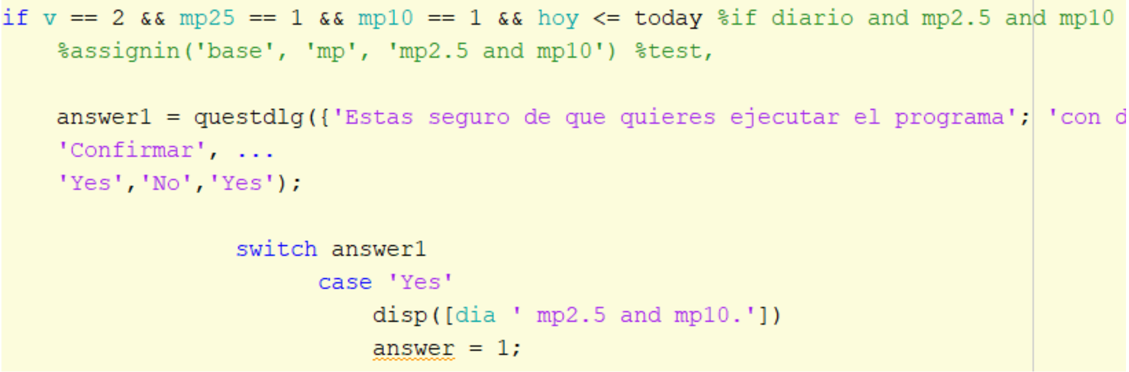
```
dia = dd + "-" + mm + "-" + yyyy;
assignin('base', 'dia',dia);
%numeral format date
DateString = dia;
formatIn = 'dd-mmm-yyyy';
hoy = datenum(DateString,formatIn);
assignin('base', 'hoy', hoy);
```

Image 26. Composing the variable *dia* (date) from day, month and year. Then the date is transformed in serial number inside the variable *hoy* and assigned to the workspace.

The last needed variables of the GUI are for the quality of the air. The variables are named for *mp25* and *mp10*. They are already chosen in radio buttons (1 = yes, 0 = no), because most common way to execute the program is with the both values. The radio buttons can be re-clicked to deselect.

The *callback* function under the Push Button *RUN* consist from six different if-functions. They are all the variations of the different documents, which are needed and possible to execute from the data. These are:

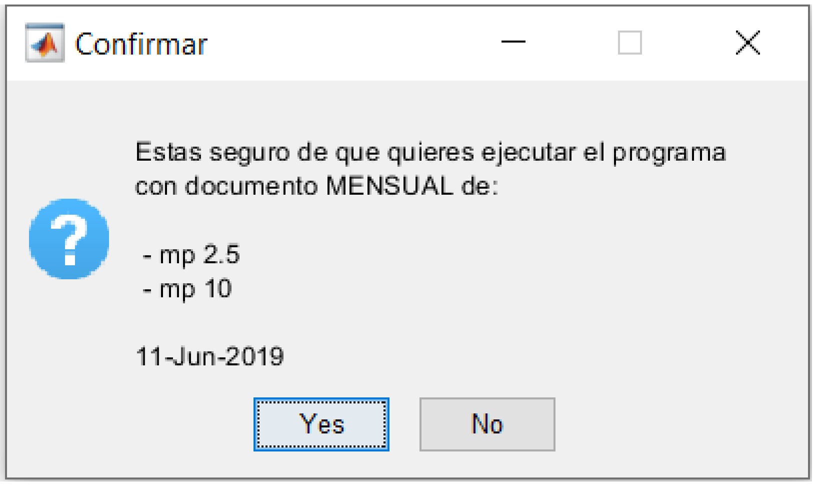
- Daily document with PM2.5 and PM10
- Daily document with PM2.5
- Daily document with PM10
- Monthly document with PM2.5 and PM10
- Monthly document with PM2.5
- Monthly document with PM10.



```
if v == 2 && mp25 == 1 && mp10 == 1 && hoy <= today %if diario and mp2.5 and mp10
    %assignin('base', 'mp', 'mp2.5 and mp10') %test,

    answer1 = questdlg({'Estas seguro de que quieres ejecutar el programa'; 'con d
    'Confirmar', ...
    'Yes','No','Yes');

                 switch answer1
                       case 'Yes'
                           disp([dia ' mp2.5 and mp10.'])
                           answer = 1;
```

Image 27.	If-function to execute the daily document with the both values of the air quality.

In Image 27. is the first if-function of the *Push button callback*. The function is for "Daily document with PM2.5 and PM10". The last condition in if, *hoy <= today*, checks that the day is not future day (the data from the future days does not exist). When pushing *RUN* and all the terms are correct, the application pop-ups a question dialog box, *questdlg*. The question dialog box asks from the user, "Are you sure that you want to execute the monthly/daily document with these data's and this day". The confirmation is made for error prevention.



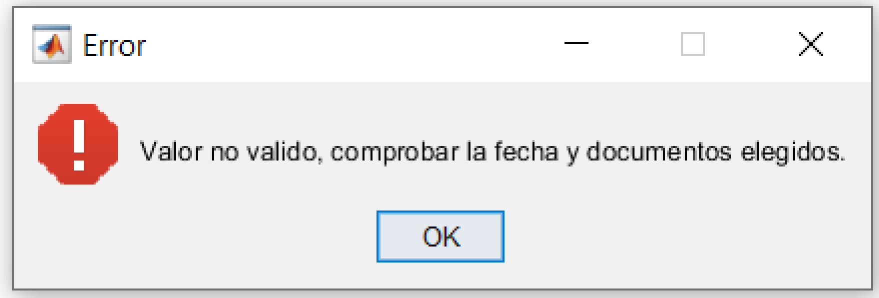
Estas seguro de que quieres ejecutar el programa
con documento MENSUAL de:

- mp 2.5
- mp 10

11-Jun-2019

Image 28.	When everything is okay, the application asks confirmation.

The program can also include errors, which are took into the account in building the application. The *else* of Run button's if-function can consist from two reasons. The possible errors in program are wrong date (future) and when both variables *mp25* and *mp10* are 0, unselected. Error opens as an error message box when trying to run the program.



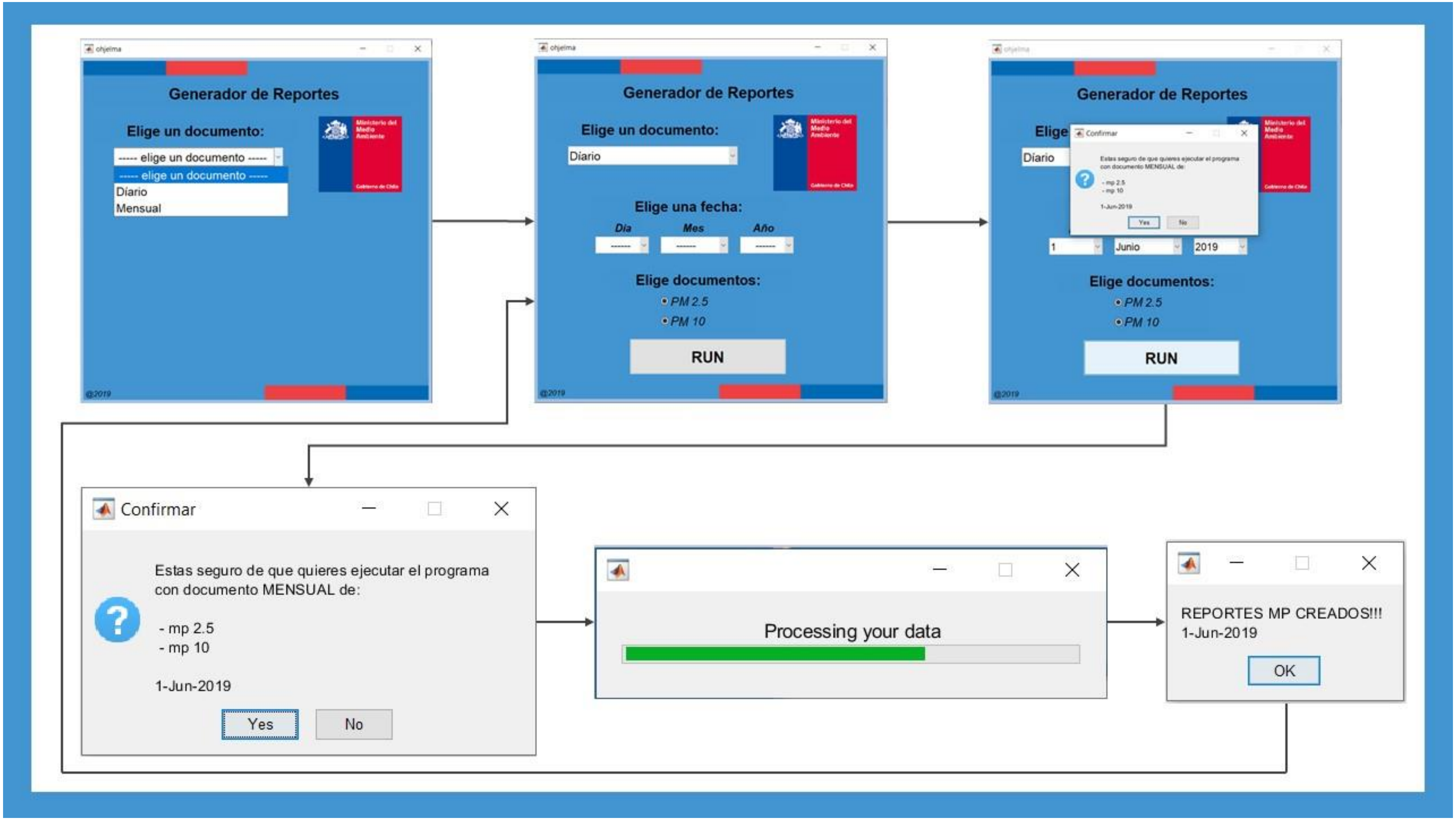
Valor no valido, comprobar la fecha y documentos elegidos.

Image 29. “Invalid value. Check the date and chosen documents”. Error in the selections when clicking the button *RUN*.

When the user does not make any errors in choosing the variables and answers to the question dialog box *“Yes”*, the program continues to execute the documents. In the User Interface, the application opens a *waitbar*, to show the user the progress of the program. The *waitbar* tells the user about the progress of the application when running. It is important for user to know in which part of the program it goes, because the running of the program takes several minutes in total. Inside the program, the program is made to run the old existing programs, which have been used earlier to make same reports.

When the *waitbar* goes to the end and the program has totally executed, the application pop-ups a message box to notify that the documents for the set day are created. After closing the message box, the application returns for the main window. After that the user can set different specs to execute other documents or if the user is ready, to close the application.

Image 30. shows the workflow of the application when the running goes correctly. In every case when the program is run, it returns to the main window of the application. In the situations when there are errors, with a date or deselecting both documents, the application also returns to the main window after closing the error message (Image 29).



Estas seguro de que quieres ejecutar el programa con documento MENSUAL de:

- mp 2.5
- mp 10

1-Jun-2019

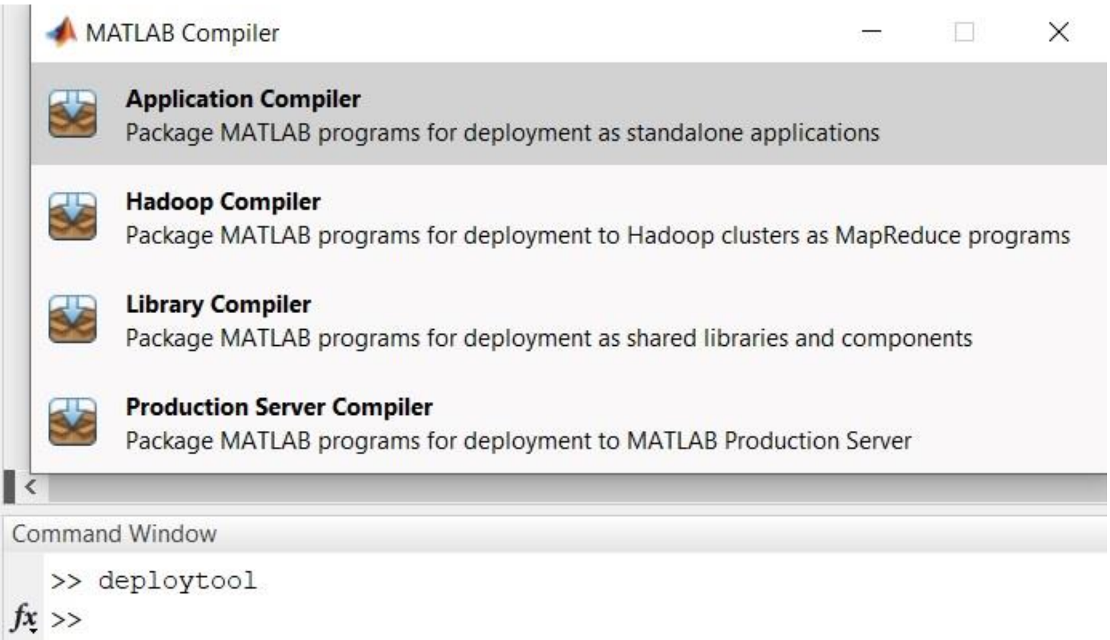
Processing your data

REPORTES MP CREADOS!!!
1-Jun-2019

Image 30. Workflow of successful executing of the application. After the reports are ready, the application returns to the main window with the selected values.

4.4 Compiling GUI to the application

With MATLAB it is possible to make standalone applications which are possible to use in computers that do not have MATLAB installed. The applications made with MATLAB can be run in Windows, Linux and Mac operating systems. Compiling a program to the application happens trough Application Compiler, which is an environment to build applications. In Application Compiler the files of the program are packed with the information and desired settings to standalone application. Accessing to Application Compiler happens through MATLAB's App's tab, or by writing *deploytool* to the command window and choosing "Application Compiler". (MathWorks: Standalone Applications 2019)



MATLAB Compiler

Application Compiler
Package MATLAB programs for deployment as standalone applications

Hadoop Compiler
Package MATLAB programs for deployment to Hadoop clusters as MapReduce programs

Library Compiler
Package MATLAB programs for deployment as shared libraries and components

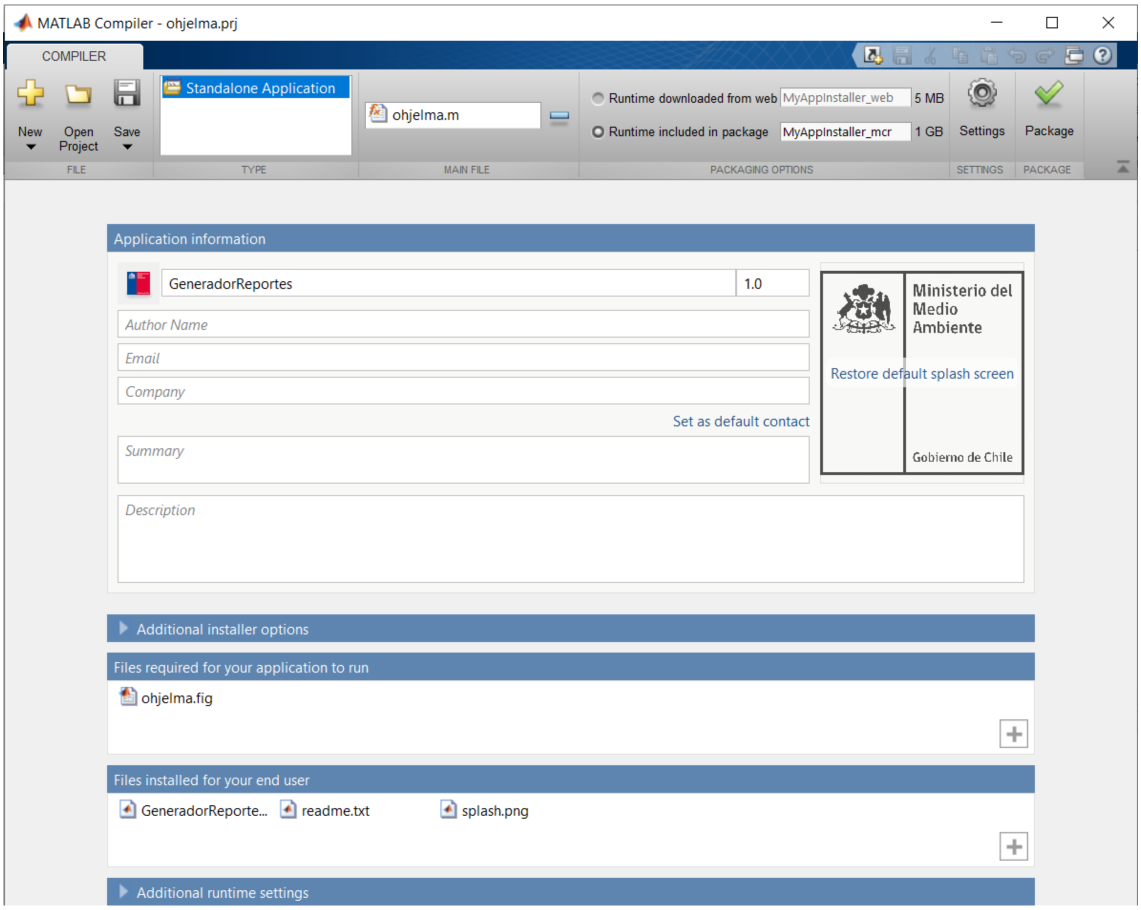
Production Server Compiler
Package MATLAB programs for deployment to MATLAB Production Server

Command Window

```
>> deploytool
>>
```

Image 31. Opening the application with the command *deploytool* to make standalone applications.

After entering to the application compiler, in the tab where reads *Add main file* is added the M-file of the program. Next to it on the right side, are selected the packaging options of the application to choose, whether the package includes MATLAB Runtime installer in the package or no. From the settings at the tab is possible to set additional parameters and select Log and Output Folders for package. From the button Package the compiler generates the packaged application. (MathWorks: Create Standalone Application from MATLAB 2019)



MATLAB Compiler - ohjelma.prj

Standalone Application

ohjelma.m

○ Runtime downloaded from web MyAppInstaller_web 5 MB

● Runtime included in package MyAppInstaller_mcr 1 GB

Application information

GeneradorReportes 1.0

Author Name

Email

Company

Summary

Description

Additional installer options

Files required for your application to run

ohjelma.fig

Files installed for your end user

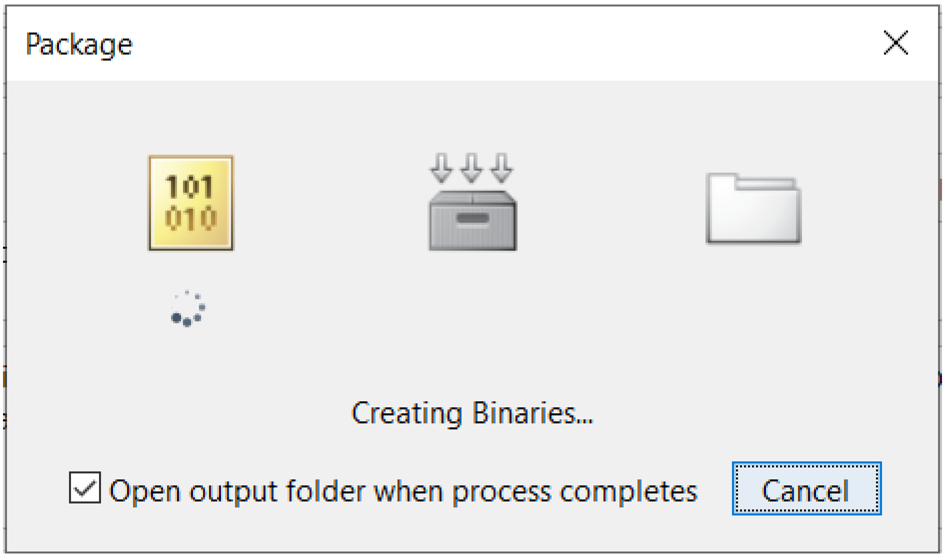
GeneradorReporte... readme.txt splash.png

Additional runtime settings

Image 32. Customizing the packaged application.

As the application of the project is made with GUIDE, the figure (F-file) of the application is added externally to the part, *Files required for your application to run*. After these steps it is possible to package a program and a figure to an application, but it would be useful to add some information for the app and enhance its appearance, by adding in application information the icon of the application and the splash screen (the screen which opens when opening the application and loading). In the application these are changed for the logos of The Ministry of The Environment of Chile, to give the “Ministerial design” which is one of the requirements.

Other information which are added in the application information are the name of the application, version (1.0 default), the name of the developer, e-mail address for support, name of the company, brief summary about the application and detailed description of the application. These information’s are mainly optional and shown almost only in the application installation. Also, in the field *Additional installer options*, are possible to give some installation notes which are shown when the installer have successfully installed the package files to the target system. It is also possible to set custom logo for the installation and change the default installation folder. (MathWorks: Customize an application 2019)

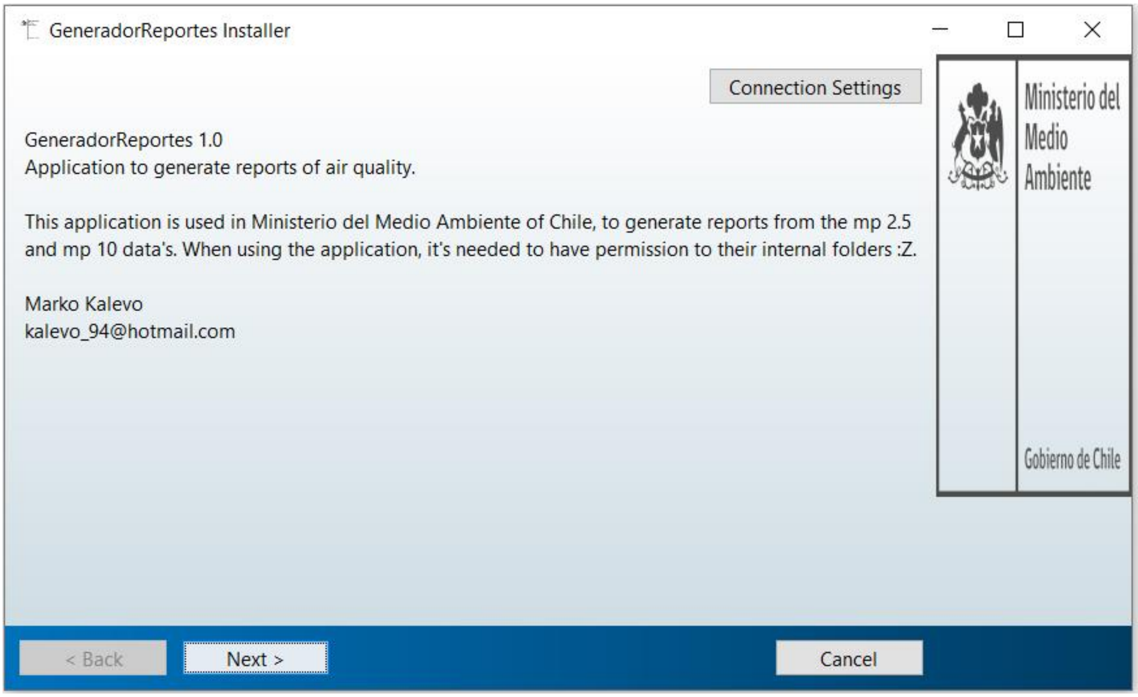


Creating Binaries...

☑ Open output folder when process completes

Image 33. Packaging the application.

When everything is ready, the needed settings and information are correct, it is time to package the program to an application. The packaging is executed from the button *Package* in the tab of the compiler. After the packaging is ready, the application is ready to run individually from the computer. To install for another computer, the package contains a folder for redistribution and an installer of the application is situated there. The computer which does not have the MATLAB software need to download MATLAB Runtime to run MATLAB made applications or components. In this case, the MATLAB Runtime installer is situated in the folders as it is selected when packaging (Image 32.). (MathWorks: MATLAB Runtime 2019)



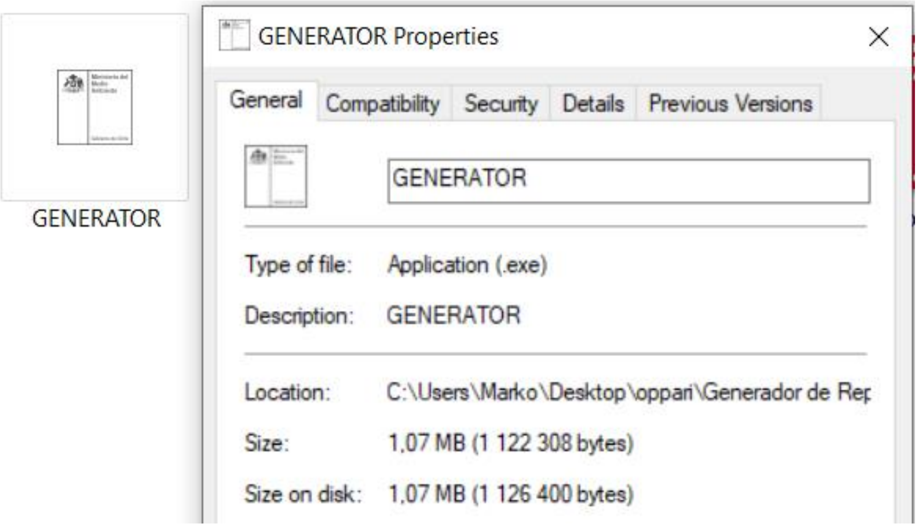
GeneradorReportes 1.0
Application to generate reports of air quality.

This application is used in Ministerio del Medio Ambiente of Chile, to generate reports from the mp 2.5 and mp 10 data's. When using the application, it's needed to have permission to their internal folders :Z.

Marko Kalevo
kalevo_94@hotmail.com

Image 34. Installing the application.

When opening the installer, the steps are simple by following the instructions and choosing the folder for saving the application. The application installation is possible to cancel while it. After that, the application can be opened and run. The application looks exactly same as when running it in MATLAB before packaging excepts the splash screen of the application (Image 30.).



GENERATOR

Type of file: Application (.exe)

Description: GENERATOR

Location: C:\Users\Marko\Desktop\oppari\Generador de Rep

Size: 1,07 MB (1 122 308 bytes)

Size on disk: 1,07 MB (1 126 400 bytes)

Image 35. Application and Properties.

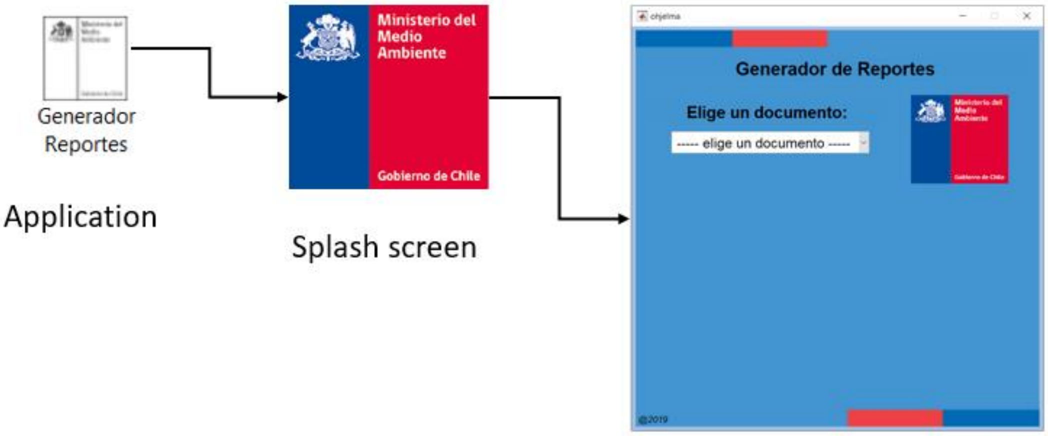


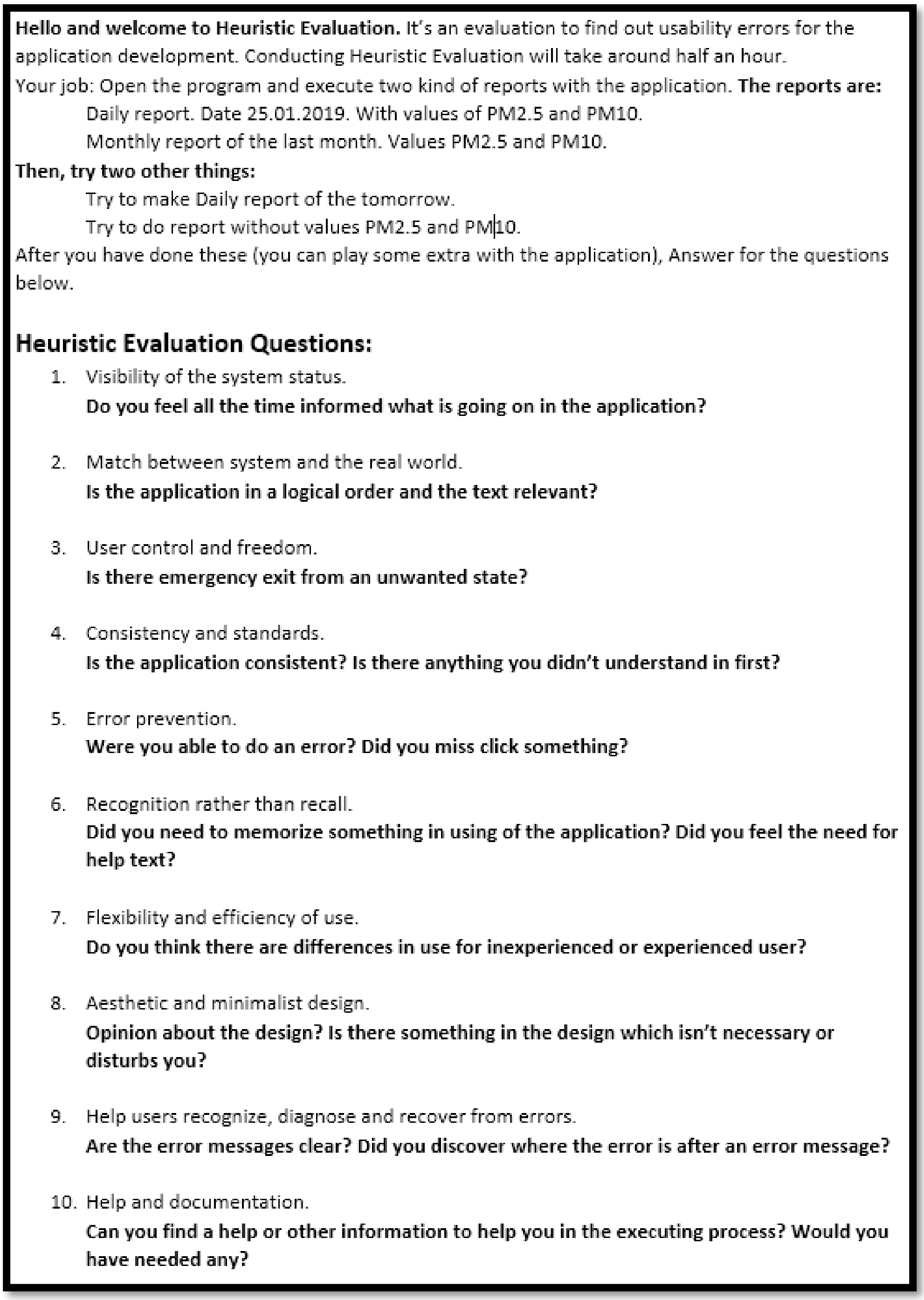
Image 36. Opening of the application. The application itself with an icon. When opened, the application pop-up's a splash screen for loading time. After that opens the application, which works similarly than in the Image 30.

4.5 Usability inspection with Heuristic Evaluation

Heuristic Evaluation is an inspection method which is used to discover errors in the application development. It is explained in the theory (3.4 Heuristic Evaluation) and it consist of 3 steps; planning, conducting and after.

The planning started by deciding which heuristic set to use and choosing evaluators. In this case, the evaluators are not chosen from the end users which would be more optical, but from a people with relevant studies and background. The Evaluators consists from ICT Specialist, Bachelor and Master student of Industrial Management and Graphic Designer. The heuristic set is the same as in the Image 15. Heuristics are modified with questions to help evaluators to understand better the ten sections.

The heuristic evaluation is conducted by meeting the evaluators individually. Evaluation of every person is carried out with 30-60-minute meet-up, which starts from explaining the task, scenario and goals of the evaluation process. The heuristic evaluation and the task are explained for the evaluators. The user test is planned to be with a minimal help for the user to test the application's easiness to use. Because the application is in Spanish, the evaluators are supported with the language if necessary.



Hello and welcome to Heuristic Evaluation. It's an evaluation to find out usability errors for the application development. Conducting Heuristic Evaluation will take around half an hour.

Your job: Open the program and execute two kind of reports with the application. **The reports are:**

Daily report. Date 25.01.2019. With values of PM2.5 and PM10.

Monthly report of the last month. Values PM2.5 and PM10.

Then, try two other things:

Try to make Daily report of the tomorrow.

Try to do report without values PM2.5 and PM10.

After you have done these (you can play some extra with the application), Answer for the questions below.

Heuristic Evaluation Questions:

1. Visibility of the system status.
   Do you feel all the time informed what is going on in the application?
2. Match between system and the real world.
   Is the application in a logical order and the text relevant?
3. User control and freedom.
   Is there emergency exit from an unwanted state?
4. Consistency and standards.
   Is the application consistent? Is there anything you didn't understand in first?
5. Error prevention.
   Were you able to do an error? Did you miss click something?
6. Recognition rather than recall.
   Did you need to memorize something in using of the application? Did you feel the need for help text?
7. Flexibility and efficiency of use.
   Do you think there are differences in use for inexperienced or experienced user?
8. Aesthetic and minimalist design.
   Opinion about the design? Is there something in the design which isn't necessary or disturbs you?
9. Help users recognize, diagnose and recover from errors.
   Are the error messages clear? Did you discover where the error is after an error message?
10. Help and documentation.
    Can you find a help or other information to help you in the executing process? Would you have needed any?

Image 37. Heuristic Evaluation form. After the form is free space for external observations. Heuristics are modified with questions to help the evaluators understand better the heuristic. (Vamsi Batchu 2018.)

After everyone have replied, the answers are compiled together and analysed. From the answers are searched the answers, where the users are not satisfied, have a problem or have something to comment. The selected answers are highlighted, and the problems analysed.

Generally, the application had good feedback from the test users. The test users commented that the logic of the application is good, and they felt satisfied with the system status and the interaction between application and user. Emergency exits exists for the users and errors in selection are not possible to execute. The application makes error message when the user selects day from the future or does not choose any documents. One of the evaluators commented why there is even the possibility to push Run button with the wrong selections if it is not possible to execute. Unfortunately, with the current components of date picking it would be complicated to implement. Also, the error message box seemed to work well. The evaluators found easily the error after the error message box.

The application itself does not involve any information about the using but every evaluator seemed to perform smoothly without problems. Evaluators commented that the application is simple and clear. Also, the evaluators commented that after the error message, the error is easy to find.

“*Design okay for the usage, but it could be more appealing”*. That sentence basically describes the opinion of test users about the design of the application. The old outlook of the application is a bit far from these days, *“but does not take away from functionality”*. Outlook of the application is much defined with the choosing of building the application with MATLAB’s guide tool. Also, the colours and logos are defined by the customer. One suggestion from the evaluator is to align components to the left side in the layout, but as it is the first person who comments about that; I think the original layout is okay.

One notice from the evaluator is concerning the application status after the report is created. The application returns to the window where the selected values are shown. The notice is that the application could return to the main window, where nothing is chosen.

To make a short conclusion about the evaluation, no serious problems were found in the user interface of the application. The chosen evaluators were well picked even though real end users were missing. The evaluation is made for the front-end of the application. The proposals are reckoned for the development, but no serious problems in the functionality were found.

5 Conclusions

This last chapter provides a short overview of the project and makes some final conclusions regarding the outcome of the project.

The main objective of the project was to build a graphical user interface to the already existing MATLAB programs to help the customer to maintain a daily routine task. The requirements of the project were to build GUI or standalone application with a good workflow, where all the variables for executing are possible to choose easily and the design is made with the customer colours and logo. The Graphical User Interface of the project needed to be deployed to the individual application to work individually without MATLAB program.

To meet all these criteria and to manage the building of the graphical user interface were needed to know how to use the MATLAB and especially how to do graphical user interfaces with it. The theory of Designing User Interface and Experience supported the building by understanding basics of application design and its importance. The Heuristic Evaluation in the end was found useful and can be recommended for inspecting usability problems in the application.

The final product of the thesis is successful and well working application. The application meets the requirements and it is working properly. The application unites using of the two different programs under one program. In the testing of the application at the customer’s office, it had some unexplained problems with the smoothness of the loading bar, but otherwise it worked properly. The application prints the documents and runs the original programs successfully and the feedback from the customer’s contact person was positive.

The main benefit of the project is the application and the benefits it gives with the better workflow and its easiness to use. The other benefit of the project is that the contact person of the project also learned the basics of building graphical user interfaces with MATLAB. With that knowledge it is possible for him to make GUI’s in the future projects.

This report contains useful information for people who are building a graphical user interface with MATLAB and especially for those who are doing it for the first time. Even if not building the application with MATLAB, the report shows basically the process of building a simple application and general knowledge needed to maintain this kind of project with its different steps.

References

UN Environment. 2017. Chile takes action on air pollution. Available at: http://ccacoalition.org/en/blog/chile-takes-action-air-pollution (Accessed: 20 September 2019.)

Jussi Huttunen. 2012. Terveyskirjasto, Hengitysilman pienhiukkaset – savusta ja saasteista sairautta, Available at: https://www.terveyskirjasto.fi/terveyskirjasto/tk.koti?p_artikkeli=kol00207&p_listatyyppi=kol) (Accessed: 20 September 2019.)

Jim Sizemore, John Paul Mueller. (2014) MATLAB For Dummies. For Dummies, November 2014. E-Book. Available at: https://learning.oreilly.com/library/view/matlab-for-dummies/9781118820032/xhtml/05_9781118820100-ch01.xhtml#a1 (Accessed: 6 May 2019)

BBC Bitesize: User Interfaces. Available at: https://www.bbc.com/bitesize/guides/zwb4jxs/revision/1 (Accessed: 6 May 2019.)

MathWorks. 2015. MATLAB: Creating Graphical User Interfaces. PDF. Available at: http://www.apmath.spbu.ru/ru/staff/smirnovmn/files/buildgui.pdf (Accessed: 6 May 2019.)

Eriksson, Lasse – Hölttä, Vesa – Riihimäki, Pasi – Varso, Joonas. (2004) MATLAB 7 – Perusteet ja sovellukset. Hakapaino Oy, Helsinki 2004.

Mathworks. 2019. MATLAB: Ways to Build Apps. Web document. Available at: https://se.mathworks.com/help/matlab/creating_guis/ways-to-build-matlab-guis.html#bu7g5rv-1 (Accessed: 7 May 2019.)

Nouman Azam. 2016. “MATLAB App Designing: The ultimate Guide for MATLAB Apps”. Video Source.

MathWorks. 2019. Create a simple app programmatically. Available at: https://se.mathworks.com/help/matlab/creating_guis/about-the-simple-programmatic-gui-example.html (Accessed: 25 April 2019.)

MathWorks. 2019. Create a Simple App GUI using Guide. Available at: https://se.mathworks.com/help/matlab/creating_guis/about-the-simple-guide-gui-example.html (Accessed: 25 April 2019.)

MathWorks. 2019. Write Callbacks in App Designer. Available at: https://se.mathworks.com/help/matlab/creating_guis/write-callbacks-for-gui-in-app-designer.html (Accessed: 26 April 2019.)

Jeff Johnson. 2007. GUI Bloopers 2.0 2nd Edition. Common User Interface Design Don'ts and Dos. Morgan Kaufmann.

Joonas Virtanen. 2016. Contrast. Mistä muodostuu loistava käyttökokemus eli User Experience (UX)? Web Blog. Available at: https://contrast.fi/hyvan-kayttokokemuksen-ux-kolme-tarkeinta-elementtia/ (Accessed: 8 May 2019.)

Interaction Design Foundation. Key question in User Experience Design – Usability vs Desirability. Available at: https://www.interaction-design.org/literature/article/key-question-in-user-experience-design-usability-vs-desirability (Accessed: 15 May 2019.)

Frank Guo. 2012. UXmatters. More Than Usability: The Four Elements of User Experience, part 1. A column. Available at: https://www.uxmatters.com/mt/archives/2012/04/more-than-usability-the-four-elements-of-user-experience-part-i.php (Accessed: 15 May 2019.)

Chauncey Wilson 2013. User Interface Inspection Methods. Morgan Kaufmann Publishers.

MathWorks 2019. Standalone Applications. Available at: https://se.mathworks.com/help/compiler/standalone-applications.html?s_tid=CRUX_lftnav (Accessed: 20 September 2019.)

MathWorks 2019. Create Standalone Application from MATLAB. Available at: https://se.mathworks.com/help/compiler/standalone-applications.html?s_tid=CRUX_lftnav (Accessed: 20 September 2019.)

MathWorks 2019. Customize an application. Available at: https://se.mathworks.com/help/compiler/customizing-the-installer.html#mw_c1f6a493-8ff5-4d61-b83b-a62999a2f009 (Accessed: 20 September 2019.)

MathWorks 2019. MATLAB Runtime. Available at: https://se.mathworks.com/products/compiler/matlab-runtime.html (Accessed: 20 September 2019.)

Vamsi Batchu, 2018. 10 tips on how to conduct a perfect Heuristic evaluation. Medium. https://medium.muz.li/10-tips-on-how-to-conduct-a-perfect-heuristic-evaluation-ae5f8f4b3257 (Accessed: 2 October 2019.)