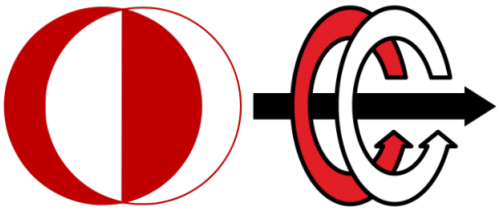


Middle East Technical University

Department of Electrical and Electronics Engineering

EE300 Summer Practice I
Report

Student Name:
Halil Temurtaş
Student ID:
2094522
SP Beginning Date:
03.07.2017
SP End Date:
28.07.2017

SP Company Name:
TÜRKSAT A.Ş.
SP Company Division:
Directorate of Satellite Programming
Supervisor Engineer:
Ömer Eren Can Koçulu
SE Contact Info:
ekoculu@turksat.com.tr

21/08/2017

Contents

1 Introduction

I have performed my summer practice in TÜRKSAT A.Ş. (Türksat Satellite Communications and Cable TV Operations Company - Türksat Uydu Haberleşme Kablo TV ve İşletme A.Ş). It is the sole communications satellite operator in Turkey. My internship lasted 20 days. Ömer Eren Koçulu, a mechatronics engineer in TURKSAT was our supervisor and he managed our internship program.

My internship started with an orientation program. The company and how works are handled were presentesd to new interns. After that, the programs and tecniques we would use in our internship and our work life were introduced. Following this introduction, a project is assigned to us as a team. Our team consisted of me and two mechatronics engineering students, Abdullah Taha İzmir and Duran Arif Göçer.

The project was about solar panels that can follow sun to increase its efficiency. In order to achieve this, we were recommended to use Raspberry Pi instead of Arduino since other team were using Arduino in their project. Moreover, we could compare the efficiency of using Raspberry and Arduino at the end. For controlling Raspberry Pi, I learnt the basics of Python and Linux enviroment. Lastly, I studied on Matlab, MS Sharepoint after finishing project.

In this report, I start with an introduction that covers what I did in my summer practice generally. Then, I continued with a company description section in which general information about TÜRKSAT's given. After this part, programs and techniques which was used throughout the summer practice is presented. After familiarizing the techniques, I gave the detailed information about the project and what I have done after project. Lastly, I finished the report with an conclusion part.

2 Description of the Company

In this chapter, I will introduce the company in five parts:

2.1 Company Name

TÜRKSAT A.Ş. (Türksat Satellite Communications and Cable TV Operations Company - Türksat Uydu Haberleşme Kablo TV ve İşletme A.Ş).

2.2 Company Location

Address-1: Ana Kampüs: Konya Yolu 40 KM. Gölbaşı/Ankara/Türkiye

Address-2: Gazi Teknokent: Bahçelievler Mahallesi, Gazi Ünv. Gölbaşı Yerleşkesi No:24, 06830 Gölbaşı/Ankara/Türkiye

Phone: +90 312 615 3000

Fax: +90 312 499 5115

2.3 General Description of the Company

Türksat Satellite Communications and Cable TV Operations Company is the sole communications satellite operator in Turkey. It was established on 21 December 1990 as a state-owned company named Türksat Milli Haberleşme Uyduları (Türksat National Communications Satellites) in Gölbaşı, Ankara Province; eventually incorporating the satellite services of Türk Telekomünikasyon A.Ş. and becoming Türksat A.Ş. on 22 July 2004. Türksat A.Ş. also owns 100% of the shares of Eurasiasat S.A.M., jointly established as a spin-off company with Aérospatiale in 1996 to manufacture and launch Turksat 2A (Eurasiasat 1) in 2001.

	Non ENgs	EE Engs	Engs
ds	14%	34	sdc
sd	23%	34	dsc

Table 1: Company Percentage

...
...
...
...

2.4 The Organizational Chart of the Company

The organizational chart of TÜRKSAT can be seen in *Figure 1*.

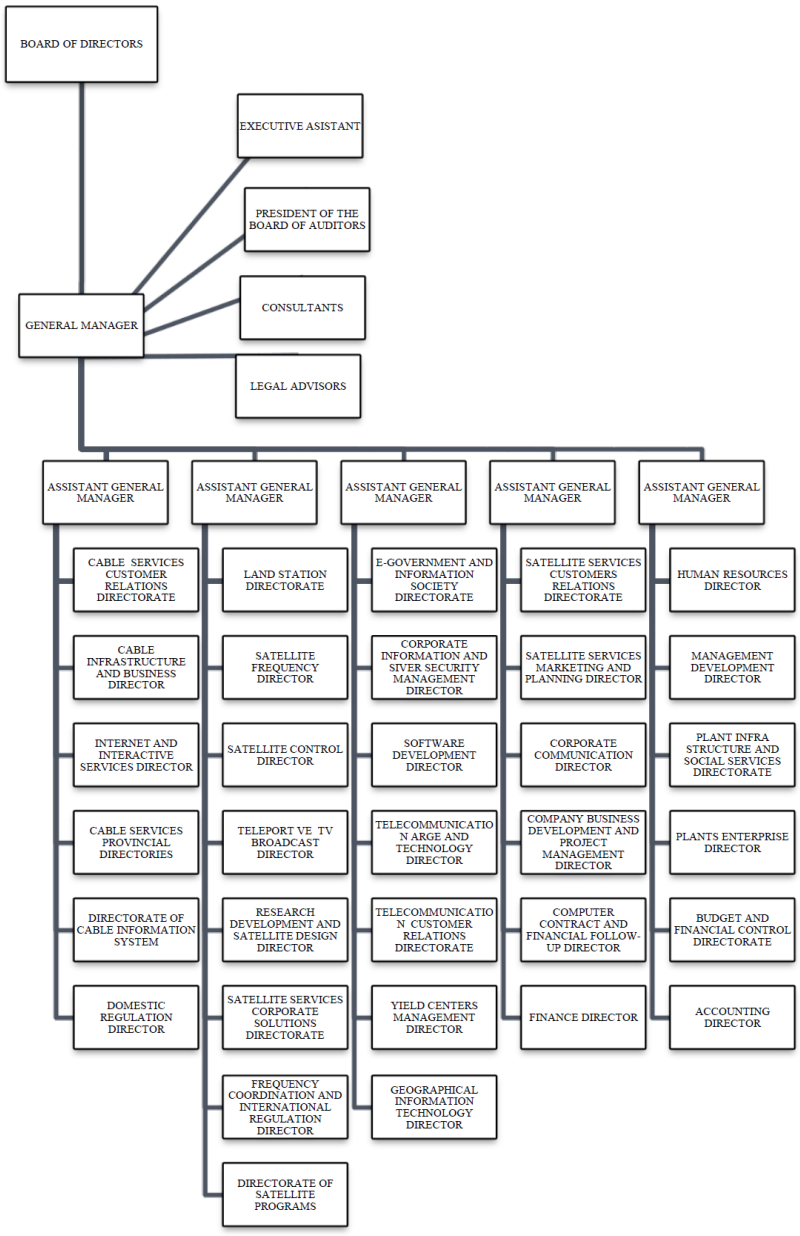


Figure 1: The Organizational Chart of TÜRKSAT

2.5 A Brief History of the Company

- **1968**

  The Chief Engineering of Satellite Telecommunications Group was established within the General Directorate of PTT.

- **August 11th, 1994**

  Turkey's Türksat 1B satellite was launched and put successfully into 42° East orbit.

- **July 10th, 1996**

  Turkey's second satellite, Türksat 1C, was launched and put into 31.3° E orbit.

- **January 11th, 2001**

  Türksat 2A (Eurasiasat 1) satellite manufactured by Eurasiasat company established in partnership with Türk Telekom and Alcatel company launched by Ariane 4 rocket from Kourou Base in South America.

- **July 22nd, 2004**

  In order to conduct satellite communication services, which was previously conducted by Türk Telekomünikasyon A.Ş., under a new company, ***Türksat A.Ş.*** was founded by Law no. 5189.

- **June 13th, 2008**

  Türksat 3A satellite launched from the French Guiana on June 13th, 2008 at 01:05 by Ariane 5 rocket and put into 42.0° East orbit.

- **February 14th, 2014**

  Turksat 4A communication satellite launched by Proton rocket from Baikonur Cosmodrome in Kazakhstan.

- **October 16th, 2015**

  Turksat 4B communication satellite launched by Proton Breeze M vehicle from Baikonur Cosmodrome in Kazakhstan and put into 50° East orbit.

3 Orientation & Useful Programs

Throughout my summer practice, I used several techniques and useful programs recommended by our supervisor.

In this section, I will explain these techniques and programs that I found very useful.

3.1 Pomodoro Technique

The Pomodoro Technique is a time management method developed by Francesco Cirillo in the late 1980s. The technique aims to increase efficiency by breaking work hours into several intervals called pomodoro. Originally 25 minutes in length, separated by short breaks, the lenght of this intervals can be changed people's personalities. For example, I have used 40 minutes lenght pomodoros, 5 minutes length short breakes and 1 hour lenght long break after 4 or 5 pomodoros. Pomodoros (tomatos in Italian) are named after the tomato-shaped kitchen timer that Cirillo used as a university student.

The tecnique is closely related to software design concepts such as incremental development and iterative and timeboxing, and has been adopted in pair programming contexts.

There are six steps in the technique:

1. *Decide on the task to be done.*
2. *Set the pomodoro timer (traditionally to 25 minutes).*
3. *Work on the task until the timer rings.*
4. *After the timer rings put a checkmark on a piece of paper.*
5. *If you have fewer than four checkmarks, take a short break (3–5 minutes), then go to step 2.*
6. *After four pomodoros, take a longer break (15–30 minutes), reset your checkmark count to zero, then go to step 1.*

A goal of the technique is to reduce the impact of internal and external interruptions on focus and flow. A pomodoro is indivisible which means it can not be interrupted. When interrupted during a pomodoro, either the other activity must be recorded and postponed (inform – negotiate – schedule – call back) or the pomodoro must be abandoned.

3.1.1 Pomotodo App

Although the creator of this technique encourages a low-tech approach that includes using a mechanical timer, paper and pencil. We have used more technological solutions called Pomotodo App in my summer practice.

The reason behind this decision was to increase efficiency even mre by using Pomotodo's some key features like built-in to-do list & category tracking system.

The stages of planning, tracking, recording, processing and visualizing are fundamental to the technique. In the planning phase tasks are prioritized by recording them in a "To Do Today" list. This enables users to estimate the effort tasks require. As pomodoros are completed, they are recorded, adding to a sense of accomplishment and providing raw data for self-observation and improvement. For that purpose, I have used Pomotodo's builtin to-do list that enables user not just tracking its work but allows user to categorise work by some cathegories. Some of my to-do list objects can be seen at *Figure 2*

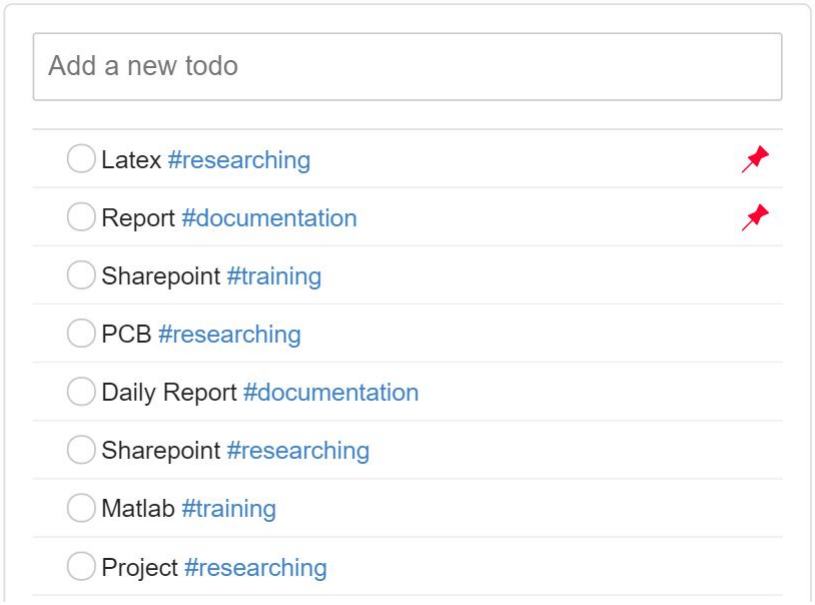


Figure 2: My To Do List in Pomotodo Web App

Figure 3: My Pomodoro History of July 12th

As can be seen at *Figure 3* , I have used some hashtags to categorise the work I have done. As can be understood from figure, 10 pomodoros were completed at July 12th. As I mentioned earlier, I have tried to use my pomodoro lenght as a 40 minutes and short breaks as 5 minutes. After 5 completed pomodoros, a long break was taken. After using this hashtags, we can ingestive our work statistic for desired times. For instance, throughout my summer practice 66% of my time was spent on training. Further statics can be seen at *Figure 4*.

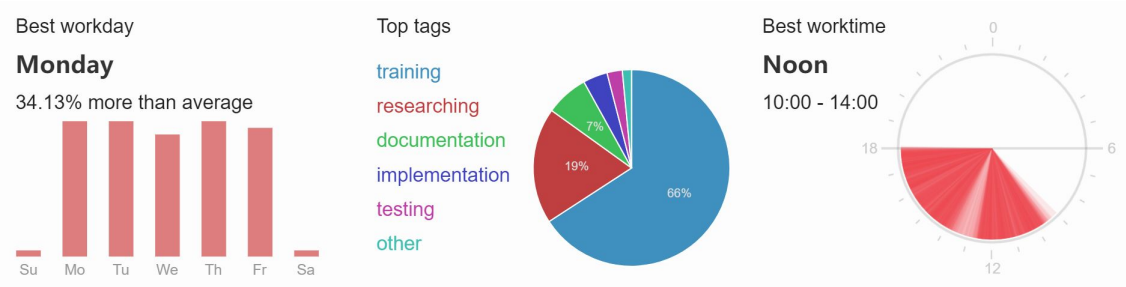


Figure 4: Some statics about my summer practice

3.2 Database Structure

A database is an organized collection of data. It is the collection of schemas, tables, queries, reports, views, and other objects. The data are typically organized to model aspects of reality in a way that supports processes requiring information, such as modelling the availability of rooms in hotels in a way that supports finding a hotel with vacancies.

Formally, a "database" refers to a set of related data and the way it is organized. Access to this data is usually provided by a "database management system" (DBMS) consisting of an integrated set of computer software that allows users to interact with one or more databases and provides access to all of the data contained in the database (although restrictions may exist that limit access to particular data). The DBMS provides various functions that allow entry, storage and retrieval of large quantities of information and provides ways to manage how that information is organized. Because of the close relationship between them, the term "database" is often used casually to refer to both a database and the DBMS used to manipulate it. Outside the world of professional information technology, the term database is often used to refer to any collection of related data (such as a spreadsheet or a card index). This article is concerned only with databases where the size and usage requirements necessitate use of a database management system.

3.2.1 Airtable

Airtable is a spreadsheet-database hybrid i.e., the features of a database are applied to a spreadsheet. The fields in an Airtable table are similar to a cell of a spreadsheet, but have types check-boxes, phone numbers, and drop-down lists, and can reference file attachments like images. Users can create a database, set up field types, add records, link tables, collaborate with a team, sort the records based on a field and publish views to external websites. When an Airtable database is created, it is automatically hosted to the cloud. The values in the fields are updated real time.

Airtable has six basic components:

Bases : All the information needed to create a project is contained in a Base. Bases can be built from existing templates provided by Airtable. In addition, they can also be built from scratch, from a spreadsheet or from an existing Base.

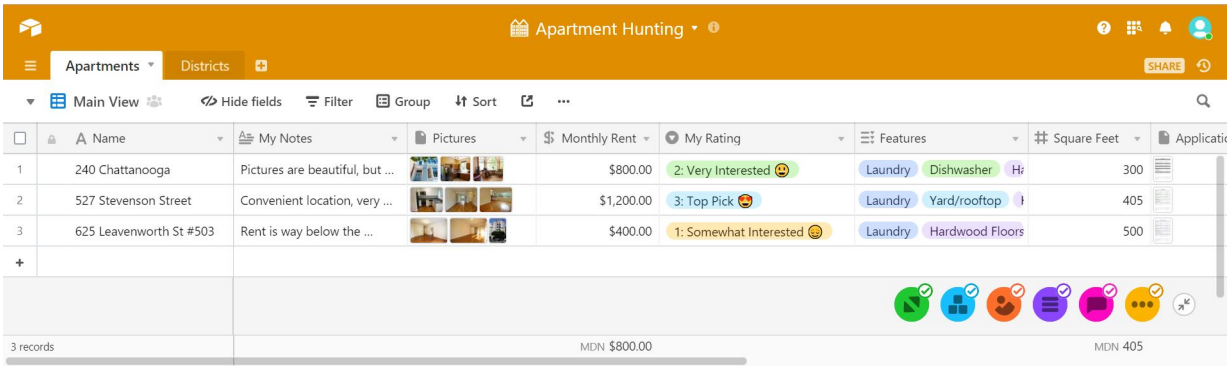


Figure 5: Apartment Hunting Base

Tables : A table is similar to an excel spreadsheet. A Base is a collection of tables.

Views : Views are how we can see a table. Views can be saved for future purposes.

Fields : Each entry in a Table is a field. They are not just restricted to hold text. Airtable currently offers 16 basic field types. These are: single-line texts, long text articles, file attachments, check-boxes, single select from drop-down list, multiple-selects from drop-down lists, date and time, phone numbers, email ids, URLs, numbers, currency, percentage, auto-number, formulae and barcodes.

Records : Each row of a Table is a Record.

Team : Team is a collection of Bases in Airtable. For example, in the adjacent restaurant template which contains all the information we need to store about the restaurants. We can have a 'Restaurants' table to store the names of restaurants along with information about their addresses, ratings, menus, etc. We can have a view to show our favourite restaurants. Each record in the Restaurants table is kept for a particular restaurant. 'Rating' can be kept as a field, to help generate 'My Apartment Hunting' view.

3.3 Wiki Pages

A wiki is a website on which users collaboratively modify content and structure directly from the web browser. In a typical wiki, text is written using a simplified mark-up language and often edited with the help of a

rich-text editor.

A wiki is run using wiki software, otherwise known as a wiki engine. A wiki engine is a type of content management system, but it differs from most other such systems, including blog software, in that the content is created without any defined owner or leader, and wikis have little implicit structure, allowing structure to emerge according to the needs of the users.

There are dozens of different wiki engines in use, both standalone and part of other software, such as bug tracking systems. Some wiki engines are open source, whereas others are proprietary. Some permit control over different functions (levels of access); for example, editing rights may permit changing, adding or removing material. Others may permit access without enforcing access control. Other rules may be imposed to organize content.

3.3.1 Confluence Wiki

Figure 6

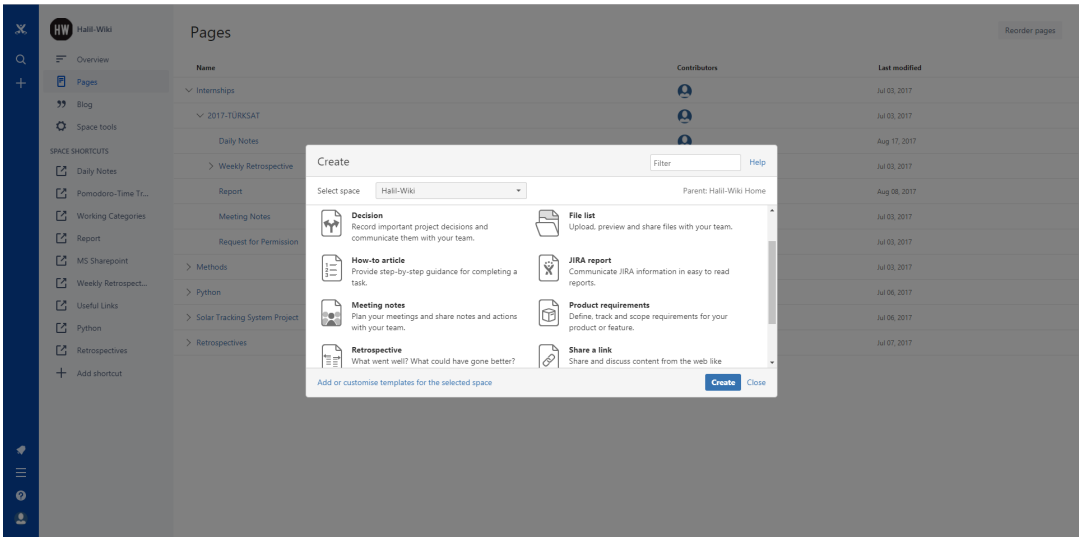


Figure 6: Confluence Wiki

There are dozens of different wiki engines in use, both standalone and part of other software, such as bug tracking systems. Some wiki engines are open source, whereas others are proprietary. Some permit control over different functions (levels of access); for example, editing rights may permit

changing, adding or removing material. Others may permit access without enforcing access control. Other rules may be imposed to organize content.

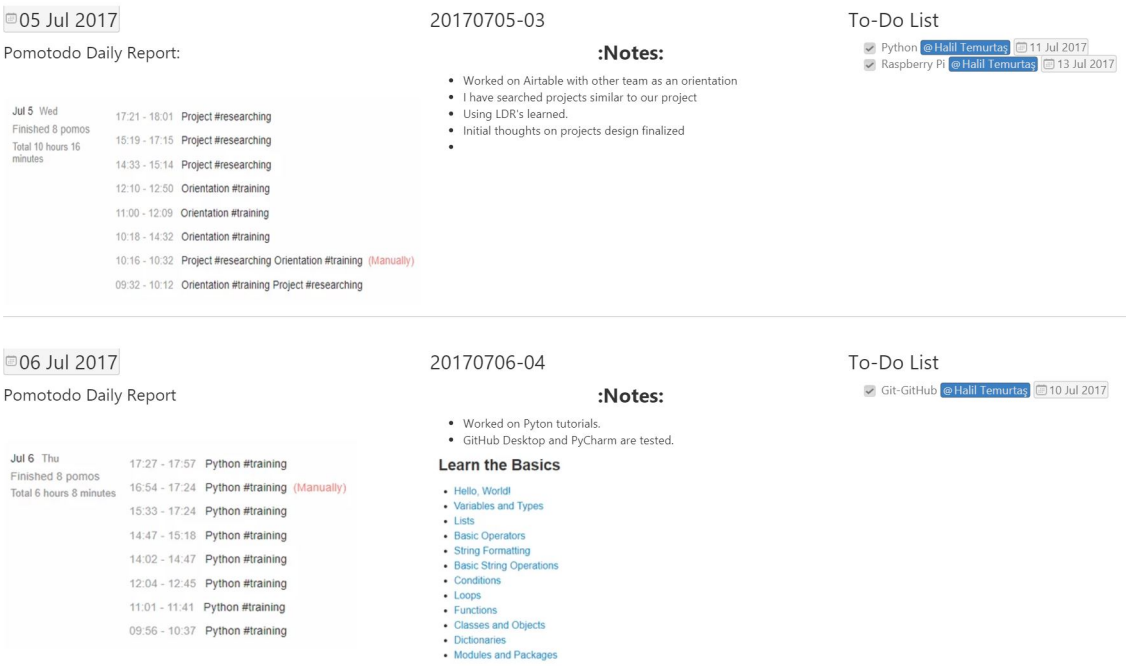


Figure 7: Body

3.4 V-Model & Agile Methodology

3.4.1 V-Model

The V-model is a graphical representation of a systems development lifecycle. It is used to produce rigorous development lifecycle models and project management models. The V-model falls into three broad categories, the German Das V-Modell, a general testing model and the US government standard.

The V-model summarizes the main steps to be taken in conjunction with the corresponding deliverables within computerized system validation framework, or project life cycle development. It describes the activities to be

performed and the results that have to be produced during product development.

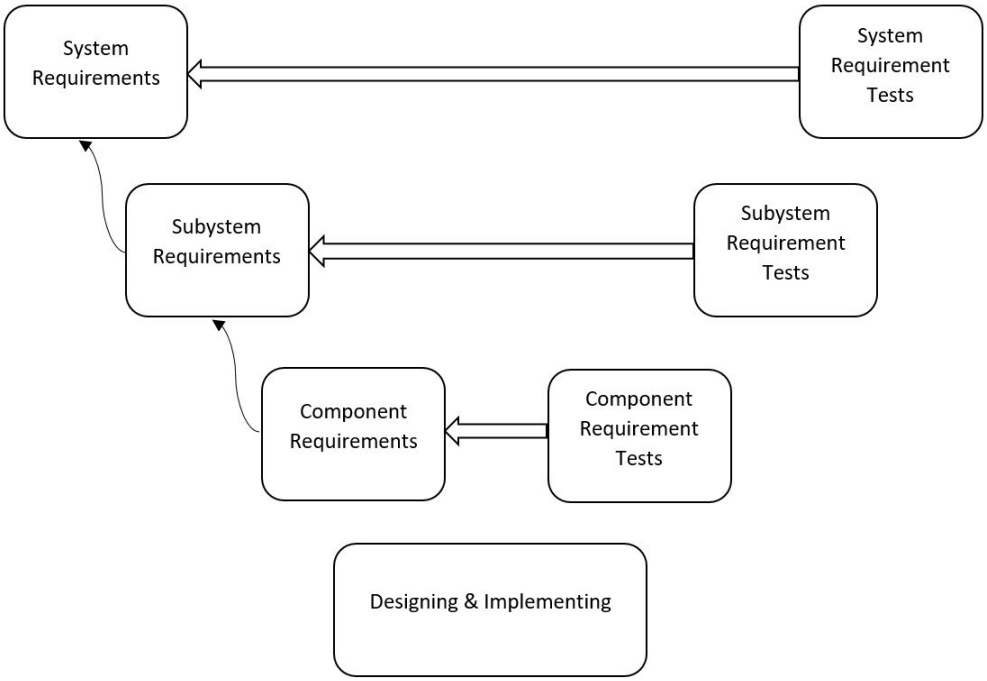


Figure 8: V-Model

The left side of the "V" represents the decomposition of requirements, and creation of system specifications. The right side of the V represents integration of parts and their validation. However, Requirements need to be validated first against the higher level requirements or user needs. Furthermore, there is also something as validation of system models (e.g. FEM). This can partially be done at the left side also. To claim that validation only occurs at the right side may not be correct. The easiest way is to say that verification is always against the requirements (technical terms) and validation always against the real world or the user needs.

In our project, as mentioned earlier in order to create the required V-model, we needed to use program that can both built the structure and track it. For that purpose we have used Airtable and its specification mentioned earlier.

3.4.2 Agile Methodology (Scrum)

Mostly used for software development, Agile describes a set of values and principles for software development under which requirements and solutions evolve through the collaborative effort of self-organizing cross-functional teams. It advocates adaptive planning, evolutionary development, early delivery, and continuous improvement, and it encourages rapid and flexible response to change. The term agile (sometimes written Agile) was popularized by the Agile Manifesto, which defines those values and principles. Agile software development frameworks continue to evolve, two of the most widely used being Scrum and Kanban. In the advice of our supervisor, We have used Scrum for our project.

3.4.2.1 Roles

As with many methodologies, Scrum also has roles for sharing the work within a team. Some of which we have used are these six:

Product Owner : The team leader, the person responsible for tracking the process.

Scrum Master : The person responsible for the correct execution of the process.

Hardware Engineer : The person or people that are responsible for designing and implementing the electrical and electronics hardware.

Software Engineer : The person or people that are responsible for creating algorithms and integration with the embedded systems.

Structure Engineer : The person or people that are responsible for integration and validation.

Test Engineer : The person or people that are responsible for system tests, subsystem tests and component tests.

3.5 Version Control with Git

Git is a version control system (VCS) for tracking changes in computer files and coordinating work on those files among multiple people. It is primarily used for source code management in software development, but it can be used to keep track of changes in any set of files. As a distributed revision control system it is aimed at speed, data integrity, and support for distributed, non-linear workflows. Git was created by Linus Torvalds in 2005

for development of the Linux kernel, with other kernel developers contributing to its initial development. Its current maintainer since 2005 is Junio Hamano. As with most other distributed version control systems, and unlike most client–server systems, every Git directory on every computer is a full-fledged repository with complete history and full version tracking abilities, independent of network access or a central server. Like the Linux kernel, Git is free software distributed under the terms of the GNU General Public License version 2.

3.5.1 Github

3.5.2 Bitbucket

Bitbucket is a web-based hosting service that is owned by Atlassian, used for source code and development projects that use either Mercurial (since launch) or Git (since October 2011) revision control systems. Bitbucket offers both commercial plans and free accounts. It offers free accounts with an unlimited number of private repositories (which can have up to five users in the case of free accounts) as of September 2010. Bitbucket integrates with other Atlassian software like Jira, HipChat, Confluence and Bamboo. It is similar to GitHub, which primarily uses Git. Bitbucket has traditionally tailored itself towards helping professional developers with private proprietary code, especially since being acquired by Atlassian in 2010. In September 2016, Bitbucket announced it had reached 5 million developers and 900,000 teams on its platform. Bitbucket has 3 deployment models: Cloud, Bitbucket Server and Data Center.

Figure 9

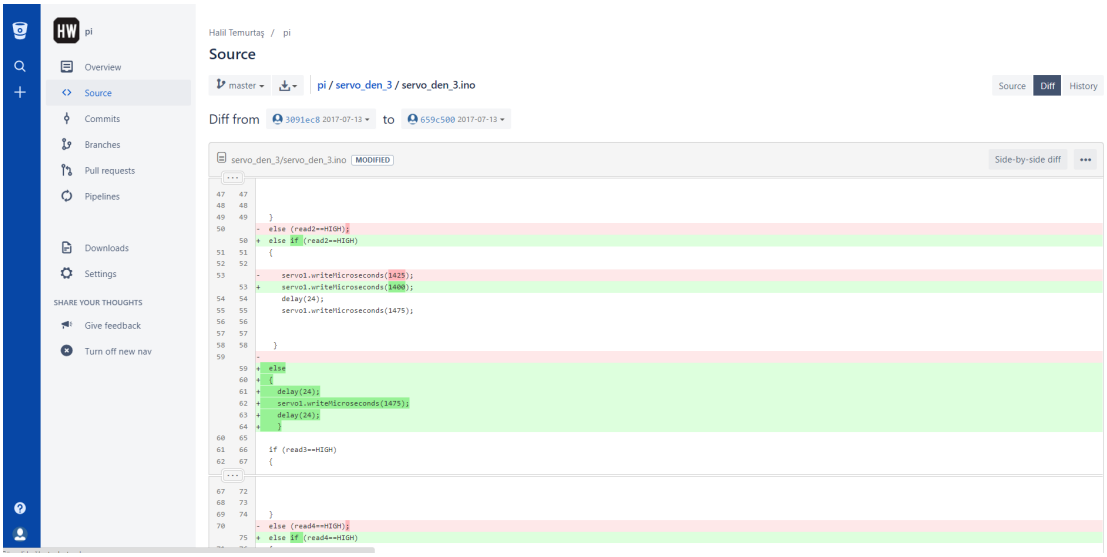


Figure 9: Bitbucket

4 Solar Tracker System Project

In my summer practice, I was assigned for a project with a team. For the project, we were expected to built a solar panel system that can follow the sun light to maxinize its efficiency.

4.1 Planning & Researching

As planning the project, we used V-model and Agile Methodology (Scrum) in order to increase efficiency and reduce time spent on the project. As mentioned earlier, using V-model required using another program. Therefore, we decided to use Airtable for tracking system requirements, subsystem requirements, tests and so on. The Interface of Airtable & System Requirements can be seen at *Figure 11*.

	Roles	Responsible Person
1	Product Owner	Halil Temurtaş
2	Scrum Master	Eren Koçulu
3	Hardware Engineer	Taha İzmir & Halil Temurtaş
4	Software Engineer	Arif Göçer & Halil Temurtaş
5	Structure Engineer	Taha İzmir & Arif Göçer
6	Test Engineer	Arif Göçer & Halil Temurtaş

Table 2: Roles

While planning the project, some roles were assigned within our team that can be seen via our HR base in Airtable in *Figure 10*. As Agile(Scrum) requires, the roles were assigned within our abilities and chooses in order to increase efficiency within the team. These roles can be seen in *Table 2*.

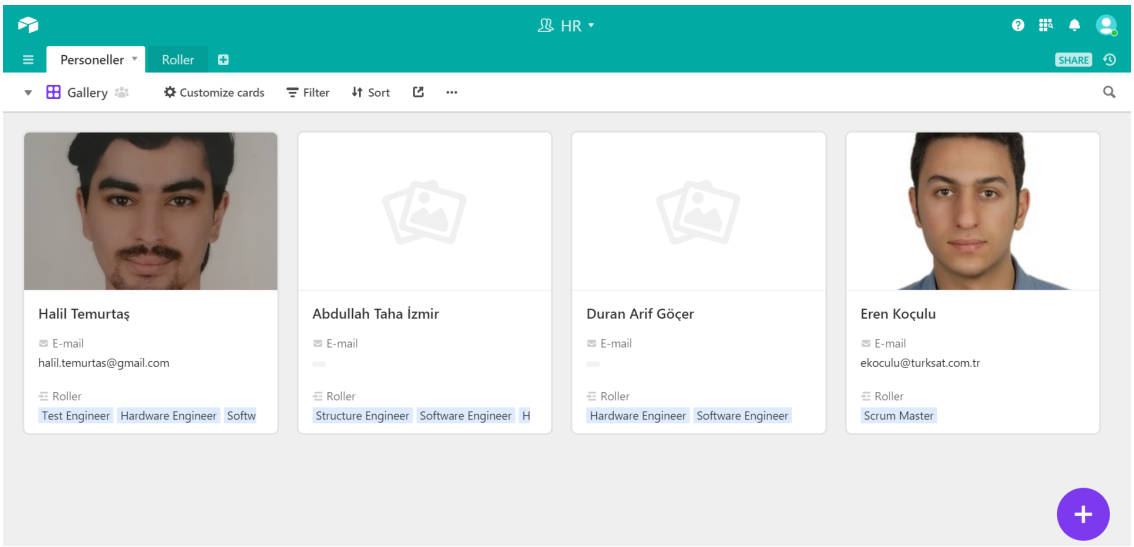


Figure 10: HR Base in the Airtabel

4.1.1 System Requirements

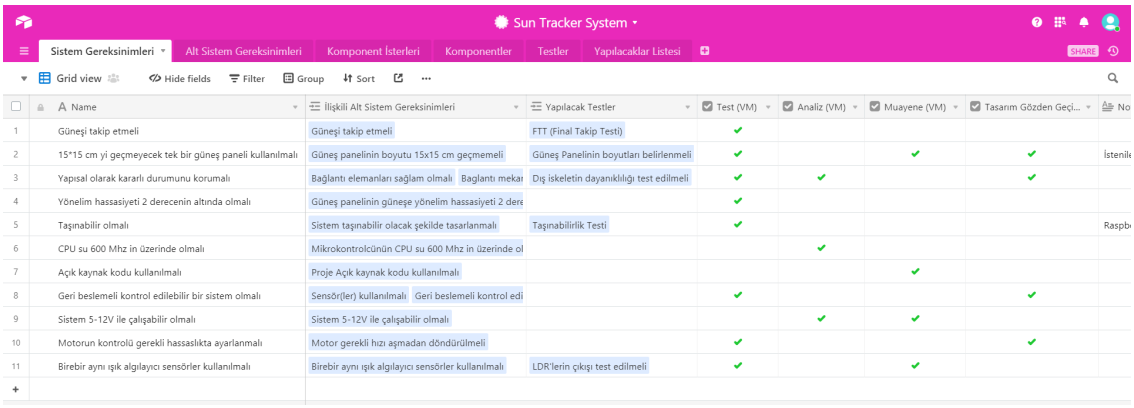


Figure 11: The Interface of Airtable & System Requirements

Constructing V-model required to specify the requirements that defines the project. For the system requirements, we considered the most basic requirements that the project must fulfil. For instance, being portable was a primary purpose for our project and it became one system requirements. From the nature of V-model, every system requirement has one or more subsystem requirement and system requirement test that will be explained later. Our project had 11 system requirements as can be seen at *Figure 11*.

4.1.2 Subsystem Requirements

As mentioned just above, every system requirement has one or more subsystem requirement that detail the requirement. As the V-model suggests, for fulfilling the system requirements, its subsystem requirements must be fulfilled first. These subsystem requirements can be considered as secondary goals that the project trying to accomplish in order to succeed its primary goals. As can be seen at *Figure 12*, we had 14 subsystem requirements for finalizing the project.

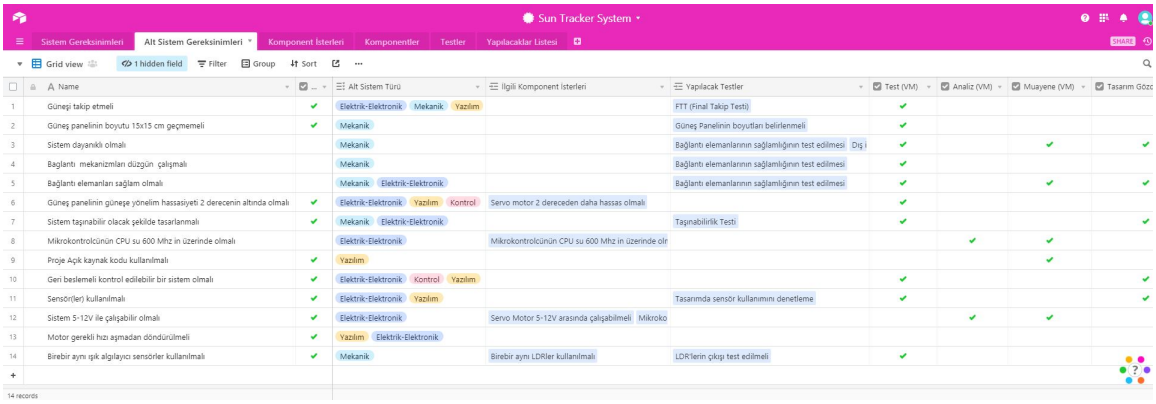


Figure 12: Body

4.1.3 Component Requirements

Figure 13

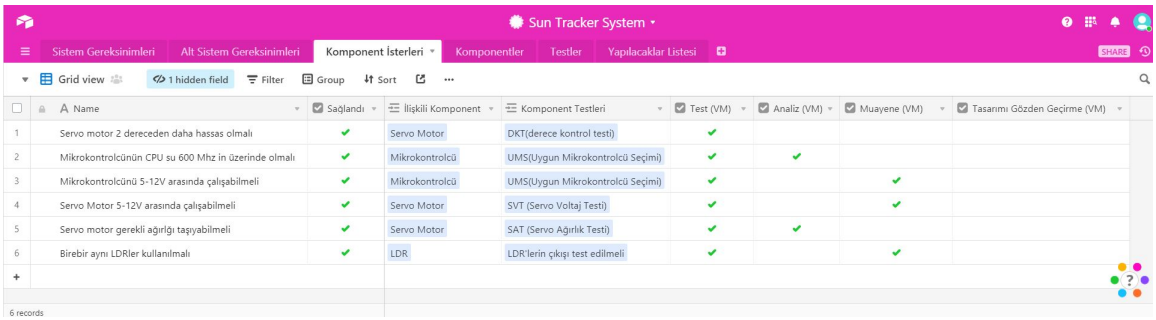
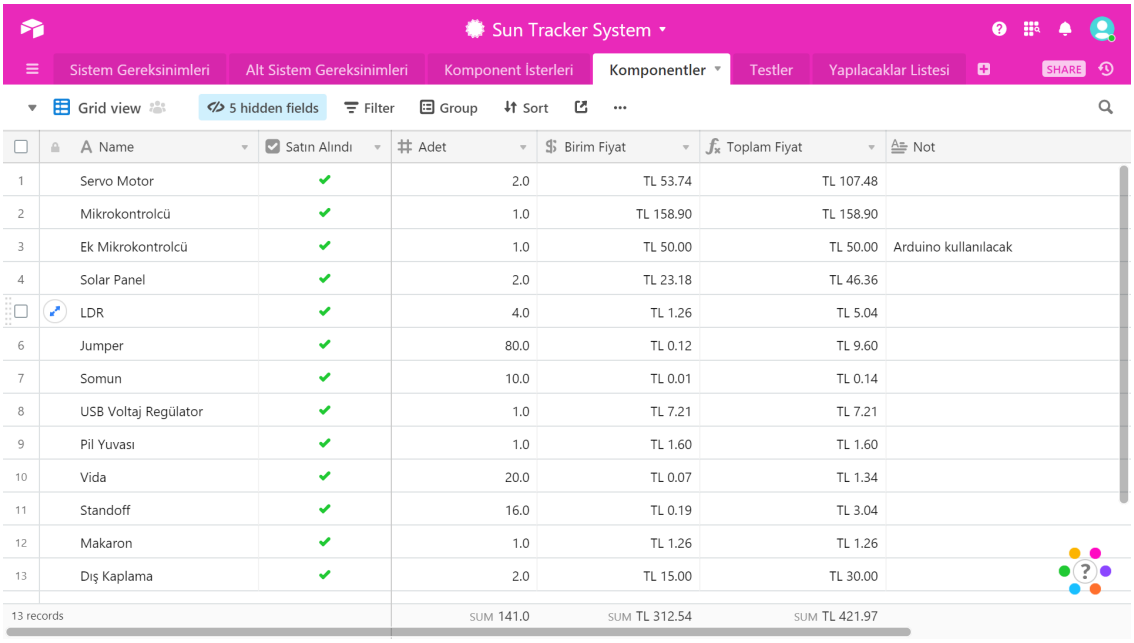


Figure 13: Body

4.1.4 Components

Figure 14



	Name	Satın Alındı	Adet	Birim Fiyat	Toplam Fiyat	Not
1	Servo Motor	✔	2.0	TL 53.74	TL 107.48	
2	Mikrokontrolcü	✔	1.0	TL 158.90	TL 158.90	
3	Ek Mikrokontrolcü	✔	1.0	TL 50.00	TL 50.00	Arduino kullanılacak
4	Solar Panel	✔	2.0	TL 23.18	TL 46.36	
	LDR	✔	4.0	TL 1.26	TL 5.04	
6	Jumper	✔	80.0	TL 0.12	TL 9.60	
7	Somun	✔	10.0	TL 0.01	TL 0.14	
8	USB Voltaj Regülator	✔	1.0	TL 7.21	TL 7.21	
9	Pil Yuvası	✔	1.0	TL 1.60	TL 1.60	
10	Vida	✔	20.0	TL 0.07	TL 1.34	
11	Standoff	✔	16.0	TL 0.19	TL 3.04	
12	Makaron	✔	1.0	TL 1.26	TL 1.26	
13	Dış Kaplama	✔	2.0	TL 15.00	TL 30.00	
13 records			SUM 141.0	SUM TL 312.54	SUM TL 421.97	

Figure 14: The components used in the project in the Airtable

4.2 Training

4.2.1 Training on Python

In order to use Raspberry Pi efficiently, I studied Python for a while from a couple of web sites. I mainly focused on Python 3 since it's more up to date than previous version. I tried different codes on Pycharm for Windows before meeting with Linux terminal and Raspberry. Pycharm is one of the most recommended Python IDE's by communities. Here are some of my very first attempts to use Python.

4.2.1.1 Basics

```
# Using Python fort he first time!!
print("Hello Intership!!!")

x = 1
if x == 1:
  # indented four spaces, indents works as brackets in C!
```

```
    print("x is 1.")
if x==3:
    print(23)

myint = 7
print(myint)  # use '#' for commenting

# A sample script that uses lists:

numbers=[]  # creates a list called numbers.
numbers.append(1) # adds '1' to numbers as first element.
numbers.append(2)
numbers.append(3)

strings=[]  # creates a list called strings.
strings.append("hello")
strings.append("world")

names = ["Ali", "Ahmet", "Ayse"]  # adds Ali, Ahmet and Ayse
                                  to names.

second_name=names[1]

print(numbers)  # prints [1, 2, 3]
print(strings)  # prints ['hello', 'world']
print("The 2nd name on the name list is %s" %second_name)
                                     # prints the second name
                                  on the names list is Ahmet!
```

```
astring = "Hello world!"

print(astring.index("o")) # prints 4, since o appears firstly
                             at 4th digit.
print(astring.count("l")) # prints 3, since l appears three
                                  times
print(astring[3:7])   # prints lo w, starting from 3rd
                                  element to 7th element (7th is
                             not included!)
print(astring[3:7:2]) # prints l, starting from 3rd element
                                  to 7th element skipping one
                                  character.
print(astring[::-1])    # prints the string reverse.
print(astring.upper())  # prints the string with upper cases.
```

```
print(astring.lower())  # prints the string with lower cases.
print(astring.startswith("Hello"))  # Returns True
print(astring.endswith("asdfasdfasdf")) # Returns False
```

4.2.1.2 Using Conditions

```
if  < statement is="" true="" > :
    < do something="" >
    ....
    ....
elif < another statement="" is="" true="" > :
     < do something="" >
    ....
    ....
else:
     < do something="" >
    ....
    ....
```

4.2.1.3 Using Loops

```
temurtas = [5, 8, 3, 6]
for halil in temurtas:
    print(halil)  # prints every element in temurtas one by
                                    one in every loop.
print(temurtas) # prints [5, 8, 3, 6]
```

```
count=0
while (count<5) :
    print(count)
    count +=1
else:
    print("count value reached %d" %(count))
```

4.2.1.4 Defining Functions

```
def sum_two_numbers(a, b):    # Defining function
    return a + b
x = sum_two_numbers(1,2)  # after this line x will hold the
                                value 3!
print("x=%s" %x)  #prints x=3
```

4.2.1.5 Defining Classes

```
class Vehicle:  # define the Vehicle class
    name = ""
    kind = "car"
    color = ""
    value = 100.00

def description(self):
    desc_str = "%s is a %s %s worth $%.2f." %(self.name,
                                    self.color, self.kind,
                                    self.value)
    return desc\_str

car1 = Vehicle()
car1.name = "Ferrari"
car1.color = "red"
car1.kind = "sport"
car1.value = 600000.00

car2 = Vehicle()
car2.name = "Jeep"
car2.color = "blue"
car2.kind = "SUV"
car2.value = 10000.00

print(car1.description()) # prints Ferrari is a red sport
                                worth $600000.00.
print(car2.description()) # prints Jeep is a blue SUV worth
                                $10000.00.
```

As I went into detail, Python is not very difficult language to learn. In fact, aside from some indent mistakes, using Python language is very simple and clean yet powerful in various applications.

4.3 Working on the Project

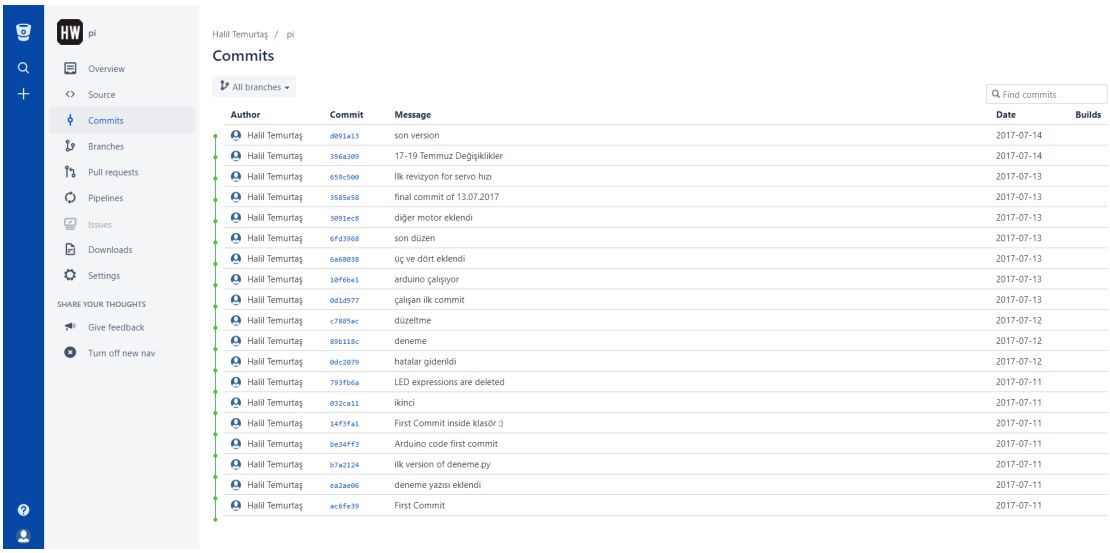


Figure 15: The Interface of Airtable & System Requirements

4.3.1 Working on Raspberry Pi

```
x==4   # first line of code on raspberry pi
if x==4
    print("evet")
```

4.3.1.1 Training on LEDs

```
import RPi.GPIO as GPIO
import time

GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)
GPIO.setup(17,GPIO.OUT)
GPIO.setup(4,GPIO.OUT)

while True:
    print "LED on"
```

```
    GPIO.output(17,GPIO.HIGH)
    GPIO.output(4,GPIO.LOW)
    time.sleep(1)
    print "LED off"
    GPIO.output(17,GPIO.LOW)
    GPIO.output(4,GPIO.HIGH)
    time.sleep(1)
```

4.3.1.2 Training on LDRs

```
from gpiozero import LightSensor, Buzzer

ldr = LightSensor(4)
ldr2 = LightSensor(17)
ldr3 = LightSensor(27)
ldr4 = LightSensor(22)

bir=ldr.value+ldr2.value
iki=ldr3.value+ldr4.value
uc=ldr.value+ldr3.value
dort=ldr2.value+ldr4.value

while True:
    print("ldr= %s" %ldr.value)
    print("ldr2= %s" %ldr2.value)
    print("ldr3= %s" %ldr3.value)
    print("ldr4= %s" %ldr4.value)
    print("bir= %s" %bir)
    print("iki= %s" %iki)
    print("uc= %s" %uc)
    print("dort= %s" %dort)
```

4.3.1.3 Training on Servo Motors

```
# Servo Control
import time
import wiringpi

wiringpi.wiringPiSetupGpio()  # use 'GPIO naming
wiringpi.pinMode(18, wiringpi.GPIO.PWM_OUTPUT)  # set pin 18
                                to be a PWM output
wiringpi.pwmSetMode(wiringpi.GPIO.PWM_MODE_MS)  # set the PWM
                                 mode to milliseconds stype
```

```
wiringpi.pwmSetClock(192)  # divide down clock
wiringpi.pwmSetRange(2000)

delay_period = 0.01

while True:
    for pulse in range(50, 250, 1):
        wiringpi.pwmWrite(18, pulse)
        time.sleep(delay\_period)
    for pulse in range(250, 50, -1):
        wiringpi.pwmWrite(18, pulse)
        time.sleep(delay_period)
```

4.3.2 Raspberry Pi Final Code

```
from gpiozero import LightSensor, Buzzer

import RPi.GPIO as GPIO
import time

GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)
GPIO.setup(23,GPIO.OUT)
GPIO.setup(24,GPIO.OUT)
GPIO.setup(25,GPIO.OUT)
GPIO.setup(8,GPIO.OUT)

ldr = LightSensor(4)# Assign the data coming from LDR1 to ldr
ldr2 = LightSensor(17)  # Assigns the data similarly
ldr3 = LightSensor(27)
ldr4 = LightSensor(22)

while True:
    bir=ldr.value+ldr2.value  # Total Readings of Top
    iki=ldr3.value+ldr4.value # Total Readings of Bottom
    uc=ldr.value+ldr3.value   # Total Readings of Left
    dort=ldr2.value+ldr4.value  # Total Readings of Right

    fark1=bir-iki;  #
    fark2=iki-bir;
    fark3=uc-dort;
    fark4=dort-uc;

    print("bir= %s" %bir)
```

```
    print("iki= %s" %iki)
    print("uc= %s" %uc)
    print("dort= %s" %dort)

    print("fark1= %s" %fark1)
    print("fark3= %s" %fark3)

    if bir>iki and fark1>0.01:
        GPIO.output(23,GPIO.HIGH)
        GPIO.output(25,GPIO.LOW)
    time.sleep(1)
    elif iki>bir and fark2>0.01:
        GPIO.output(25,GPIO.HIGH)
        GPIO.output(23,GPIO.LOW)
        time.sleep(1)
    else :
        GPIO.output(25,GPIO.LOW)
        GPIO.output(23,GPIO.LOW)
        time.sleep(1)

    if uc>dort and fark3>0.01:
        GPIO.output(24,GPIO.HIGH)
        GPIO.output(8,GPIO.LOW)
        time.sleep(1)
    elif dort>uc and fark4>0.01:
        GPIO.output(8,GPIO.HIGH)
        GPIO.output(24,GPIO.LOW)
        time.sleep(1)
    else :
        GPIO.output(24,GPIO.LOW)
        GPIO.output(8,GPIO.LOW)
        time.sleep(1)
```

4.3.3 Working on Arduino

4.3.3.1 Training on LEDs & Pins

4.3.3.2 Training on Servo Motors

```
#include <Servo.h>

Servo Servo1; // create servo named Servo1 to control a servo
int pos = 0;  // variable to store the servo position }

```

```
6  void setup()
7  {
8    Servo1.attach(9); // attaches the servo on pin 9 to the servo
       object
9  }
10
11 void loop()
12 {
13     for (pos = 0; pos <= 180; pos += 1) // goes from 0 degrees
     to 180 degrees in steps of 1 degree
14     {
15         Servo1.write(pos);  // tell servo to go to position in
     variable 'pos'
16         delay(15);  // waits 15ms for the servo to reach the
     position
17     }
18     for (pos = 180; pos >= 0; pos -= 1) // goes from 180 degrees
       to 0 degrees
19     {
20        Servo1.write(pos); // tell servo to go to position in
     variable 'pos'
21         delay(15);  // waits 15ms for the servo to reach the
     position
22     }
23 }
```

4.3.4 Final Arduino Code

```
1  #include <Servo.h>
2
3  Servo servo1;
4  Servo servo2;
5
6  int in_rasp1 =3;
7  int in_rasp2 =4;
8  int in_rasp3 =5;
9  int in_rasp4 =6;
10
11 int read1=0;
12 int read2=0;
13 int read3=0;
14 int read4=0;
15
16 void setup()
17 {
18     servo1.attach(9);
```

```
19      servo1.writeMicroseconds(1475);
20      servo2.attach(10);
21      servo2.writeMicroseconds(1475);
22
23      pinMode(in_rasp1, INPUT);
24      pinMode(in_rasp2, INPUT);
25      pinMode(in_rasp3, INPUT);
26      pinMode(in_rasp4, INPUT);
27  }
28  void loop()  {
29      read1 =digitalRead(in_rasp1);
30      read2 =digitalRead(in_rasp2);
31      read3 =digitalRead(in_rasp3);
32      read4 =digitalRead(in_rasp4);
33
34      if (read1 == HIGH)
35      {
36          servo1.writeMicroseconds(1515);
37          delay(42);
38          servo1.writeMicroseconds(1475);
39          delay(200);
40      }
41      else if (read2 == HIGH)
42      {
43          servo1 .writeMicroseconds(1425);
44          delay(24);
45          servo1.writeMicroseconds(1475);
46          delay(100);
47      }
48      else
49      {
50          delay(24);
51          servo1.writeMicroseconds(1475);
52          delay(24);
53      }
54      if (read3 == HIGH)
55      {
56          servo2.writeMicroseconds(1515);
57          delay(42);
58          servo2.writeMicroseconds(1475);
59          delay(100);
60      }
61      else if (read4 == HIGH)
62      {
63          servo2.writeMicroseconds(1425);
```

```
64            delay(24);
65            servo2.writeMicroseconds(1475);
66            delay(100);
67        }
68        else
69        {
70            delay(24);
71            servo2.writeMicroseconds(1475);
72            delay(24);
73        }
74    }
```

4.4 Implementation

4.4.1 PCB Drawing

Before beginning the implementation, our aim was to create the circuit in some PCB making software and print it. However, since the whole circuit we wanted to use was too simple for drawing and covered the whole solar panels. We decided to solder all pieces with the jumpers. First draft of our circuit can be seen at *Figure 16* and the final product which we used soldiring can be seen at *Figure 17*.

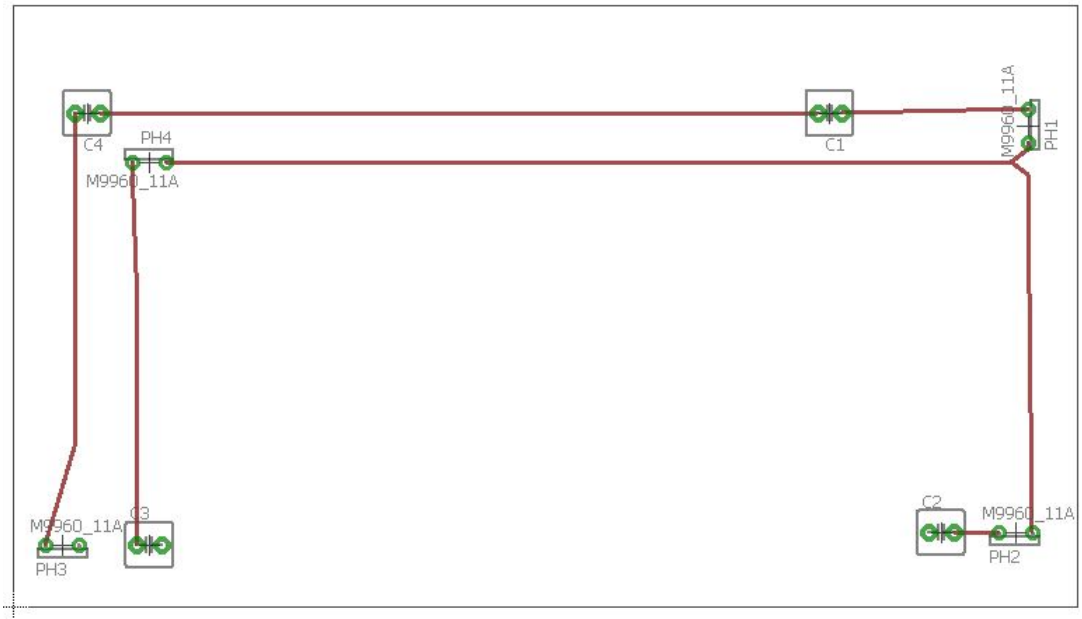


Figure 16: PCB

4.4.2 3D Drawings

...
...
...
...
...
...
...
...
...
...

4.4.3 Construction of the Body

4.4.3.1 Top Layer

Top layer can be seen at *Figure 17*

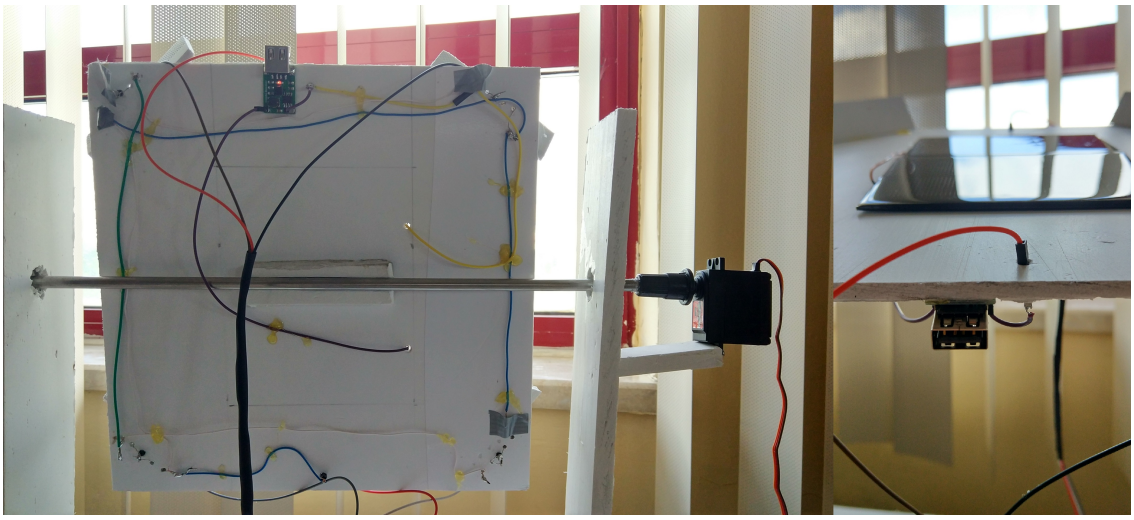


Figure 17: Top Layer

4.4.3.2 Main Body

Top main body can be seen at *Figure 18*

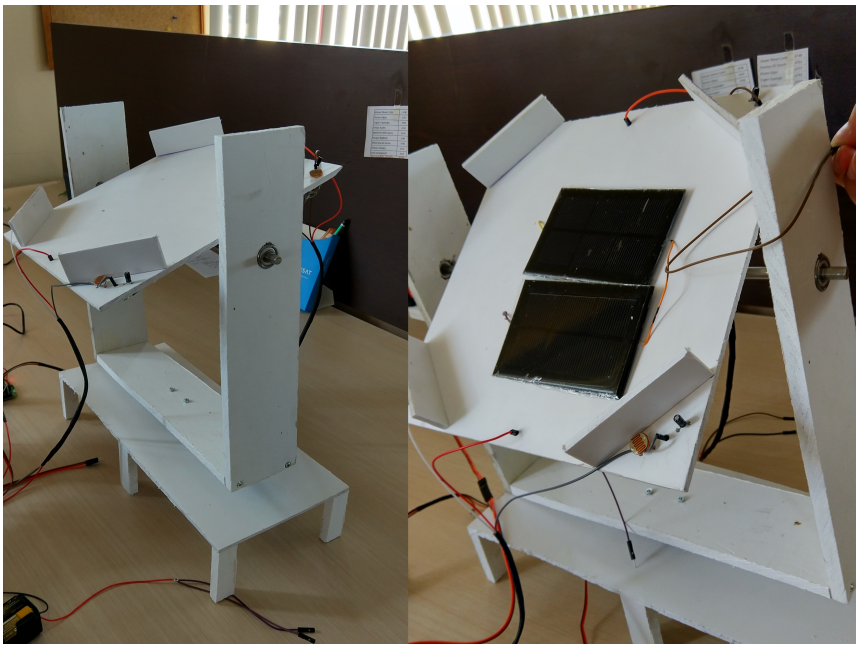


Figure 18: Body

4.4.3.3 Solar Panel

As can be understood from the project name, our project was all about solar panel. The reason behind tracking the Sun was to increase efficiency. In this part of the project, two 11 cm*7 cm length solar panel that are 1.5 V ideally is connected serial in order to get 3 Volts of potential difference between its legs. We have soldered one of this leg to ground and the other one to the input of USB voltage regulator that increases voltage to 5V. Then this potential is used in order to charge the phones and other USB powered stuff. However, due to solar panel that was not capable of producisng 1.5V and regulator that gets input between 2-5V, we was not able to produce any voltage at the output of regulator. The regulator can be seen at the right side of theat *Figure 17* and panels can be seen at *Figure 18*.

4.4.3.4 Final Body

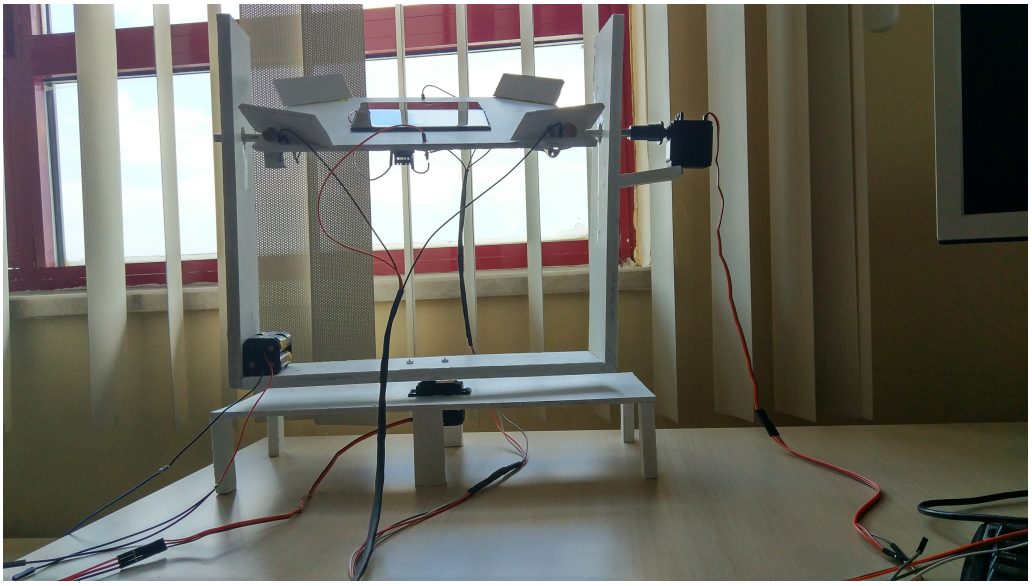


Figure 19: Final Body

After a couple of days work, we constructed the main body that can be seen at *Figure 19*, we moved on the tests which we should fulfil in order to complete the V-Model.

4.5 Tests

As V-Model suggests, we started with the component tests which decides whether the specific component requirement is completed. Since the subsystem requirements are connected to the component requirements. We have moved on to the subsystem requirement tests. Due to some problems in the project, we was not able to complete and succeed in all subsuytem tests. Therefore, most of the system requirement tests was not completed. Whole tests and the ones accomplished can be seen at at *Figure 20*.

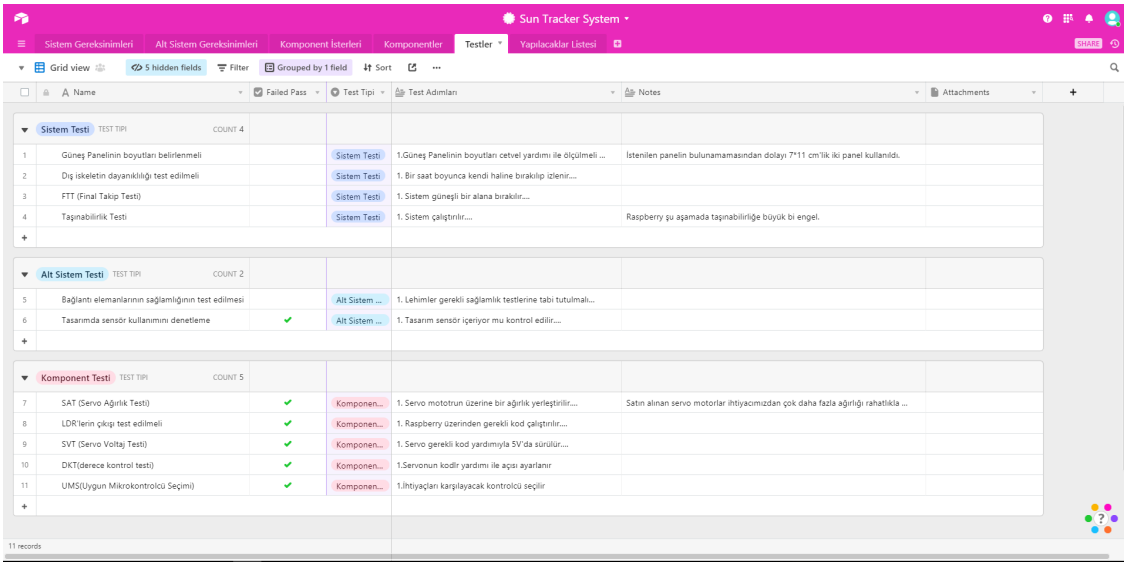


Figure 20: Tests

4.6 Project Tracking

4.6.1 Kanban

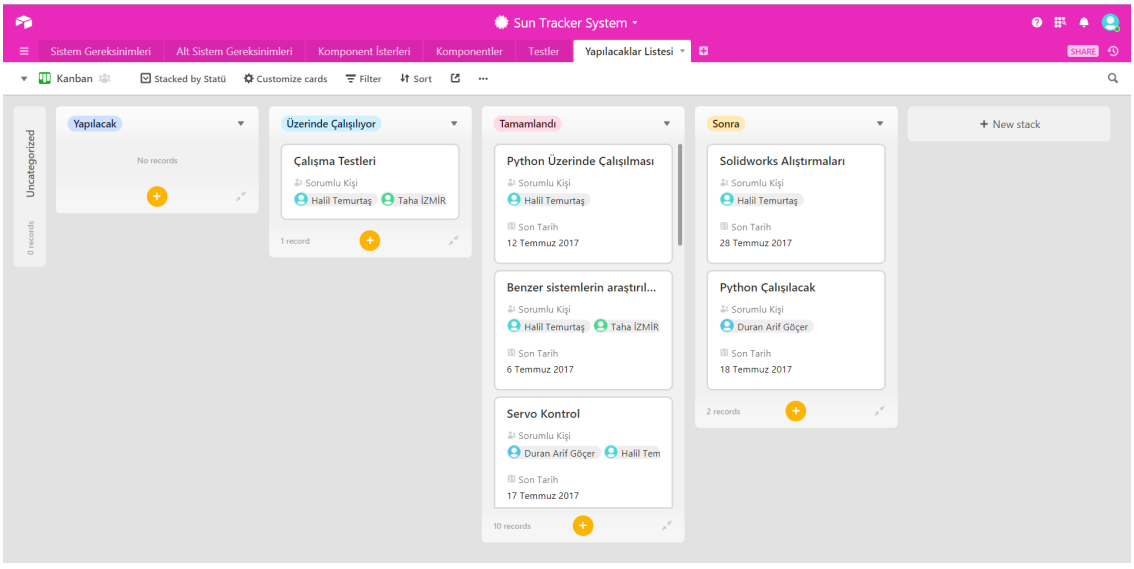


Figure 21: Body

5 After Project

5.1 Training on MATLAB

After finishing the project earlier than expected, I was asked to study for educational purposes. Firstly, PCB designing and Solidworks modelling were my priorities since I was not able to do both during the project. Due to limited time, I did not choose either. Since I know the basics, I have chosen Matlab to study on it.

5.1.1 Coursera

For that purpose, I have enrolled a course on Coursera. Coursera is.... ...
...
...

5.1.2 Outline of the Course

In the first two weeks of the course program, as can be seen from *Figure 22*, Matlab environment and basic operators were introduced. Since I know them already, I have watched the video lectures in a few hours. After that,

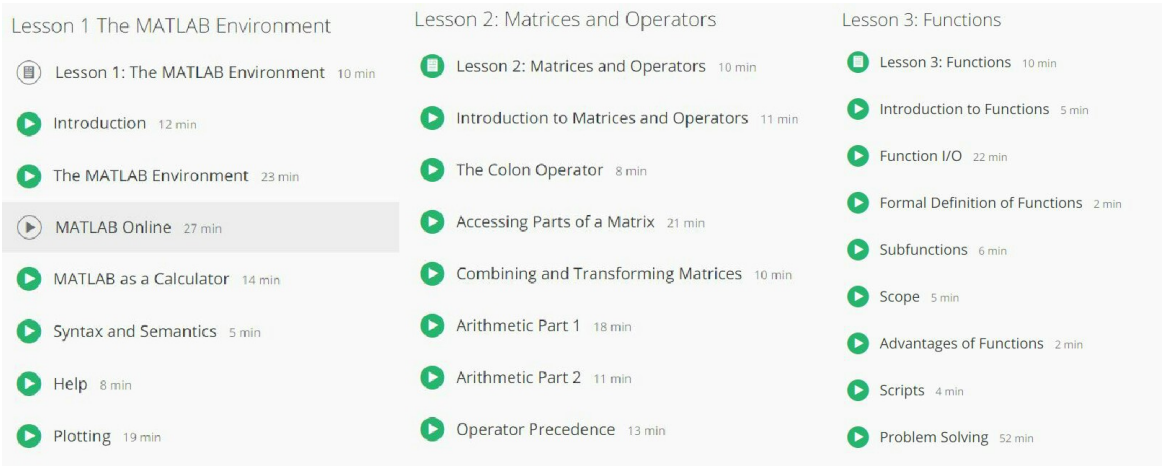


Figure 22: The Syllabus of Matlab Course for First 3 Weeks

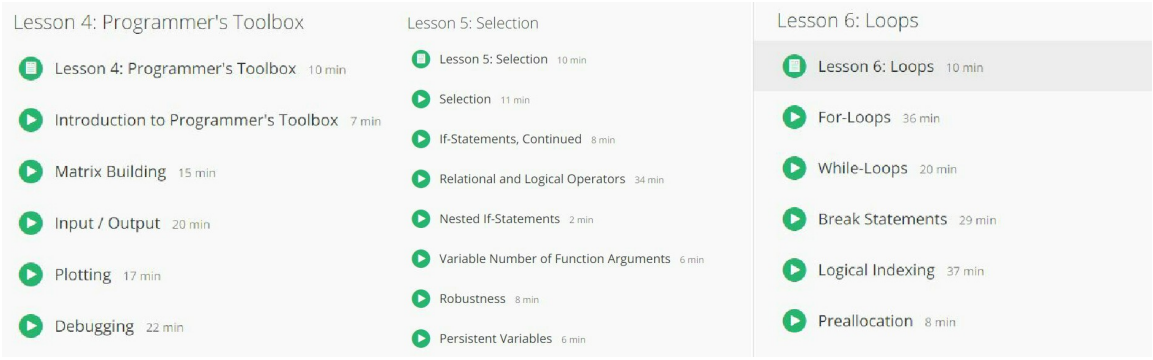


Figure 23: The Syllabus of Matlab Course for 4-5-6 Weeks

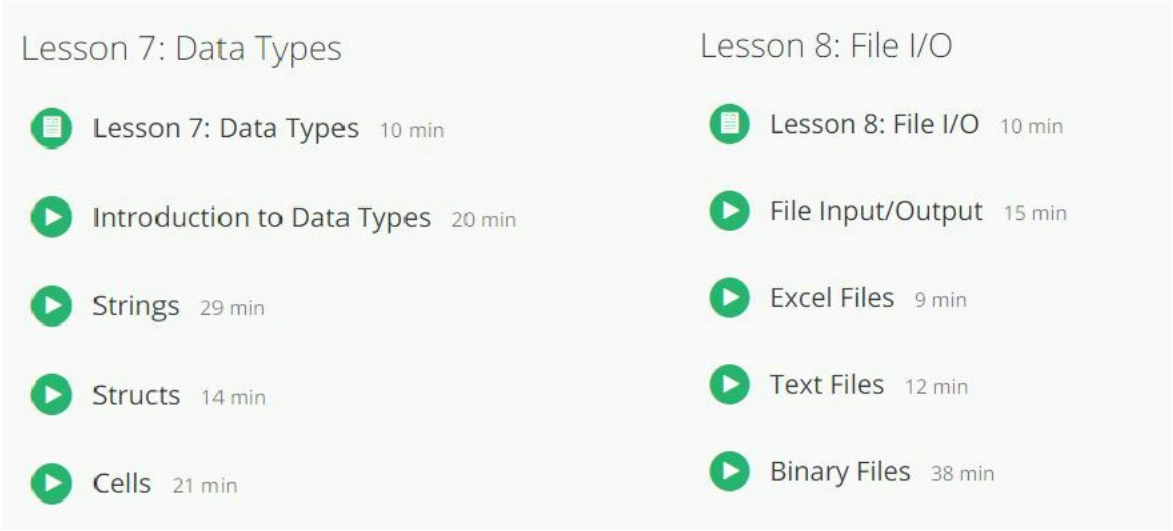


Figure 24: The Syllabus of Matlab Course for Last 2 Weeks

5.1.2.1 Simple Sorting Code

```
function [a b c] = sort3(A)
a1 = A(1)
a2 = A(2)
a3 = A(3)

```

```
6  if a1 <= a2
7      if a2 <= a3
8          a = a1
9          b = a2
10         c = a3
11     else
12         e = a3
13         a3 = a2
14         a2 = e
15
16         if a1 <= a2
17             a = a1
18             b = a2
19             c = a3
20         else
21             w = a2
22             a2 = a1
23             a1 = w
24             a = a1
25             b = a2
26             c = a3
27         end
28     end
29 else
30     w = a2
31     a2 = a1
32     a1 = w
33     if a2 >= a3
34         e = a3
35         a3 = a2
36         a2 = e
37         if a1 <= a2
38             a = a1
39             b = a2
40             c = a3
41         else
42             w = a2
43             a2 = a1
```

```
                a1 = w
                a = a1
                b = a2
                c = a3
            end
        else
            a = a1
            b = a2
            c = a3
        end
end
end
}
```

5.2 Training on Microsoft Sharepoint

5.2.1 Microsoft Sharepoint

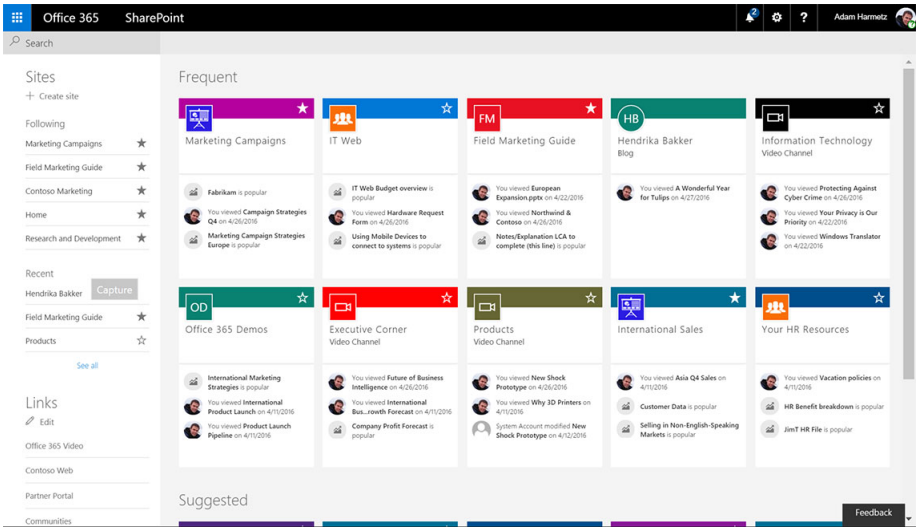


Figure 25: Body

SharePoint is a web-based, collaborative platform that integrates with Microsoft Office. Launched in 2001, SharePoint is primarily sold as a document management and storage system, but the product is highly configurable

and usage varies substantially between organizations. Microsoft states that SharePoint has 190 million users across 200,000 customer organizations.

Figure 26

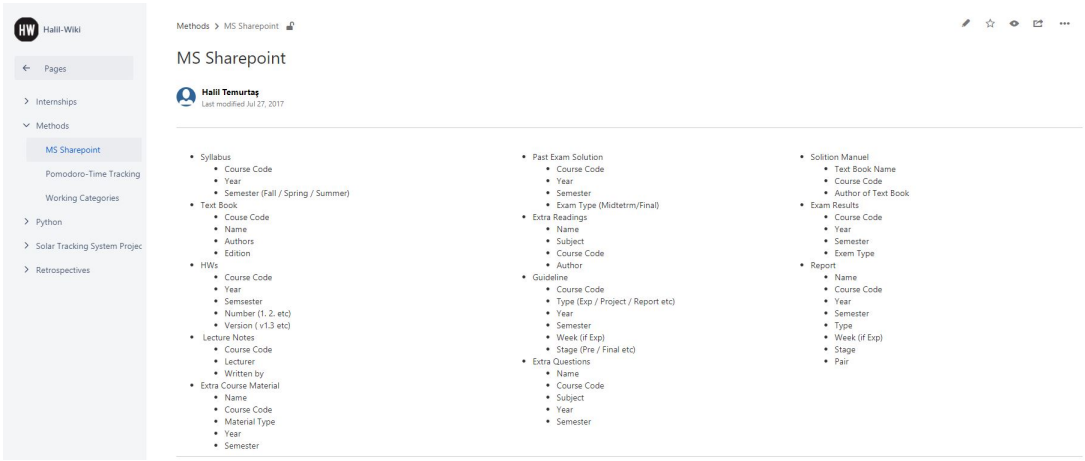


Figure 26: Body

Figure 27

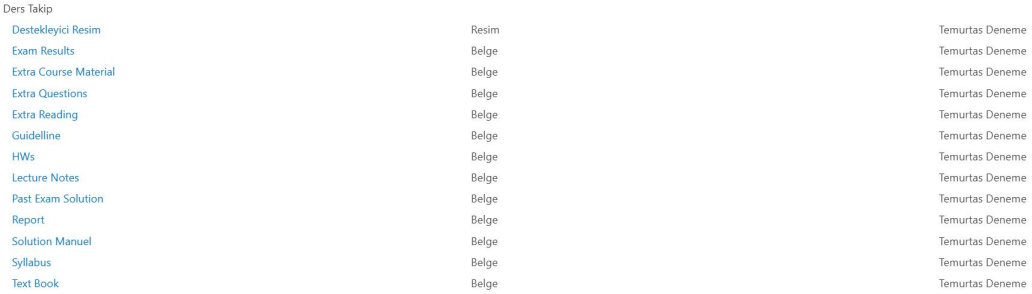
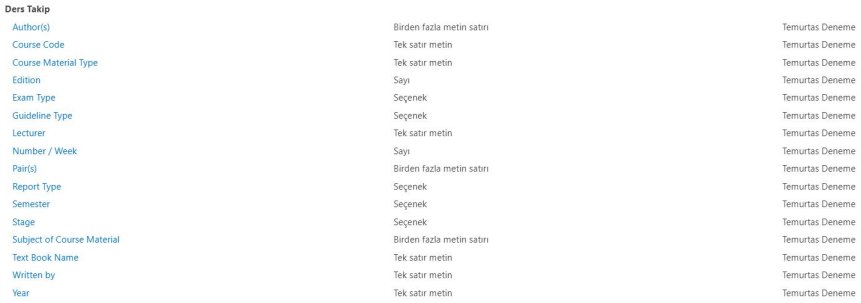


Figure 27: Body

Figure 28



Ders Takip

Author(s)	Birden fazla metin satırı	Temurtas Deneme
Course Code	Tek satır metin	Temurtas Deneme
Course Material Type	Tek satır metin	Temurtas Deneme
Edition	Sayı	Temurtas Deneme
Exam Type	Seçenek	Temurtas Deneme
Guideline Type	Seçenek	Temurtas Deneme
Lecturer	Tek satır metin	Temurtas Deneme
Number / Week	Sayı	Temurtas Deneme
Pair(s)	Birden fazla metin satırı	Temurtas Deneme
Report Type	Seçenek	Temurtas Deneme
Semester	Seçenek	Temurtas Deneme
Stage	Seçenek	Temurtas Deneme
Subject of Course Material	Birden fazla metin satırı	Temurtas Deneme
Text Book Name	Tek satır metin	Temurtas Deneme
Written by	Tek satır metin	Temurtas Deneme
Year	Tek satır metin	Temurtas Deneme

Figure 28: Body

6 Conclusion

I completed my summer practice in TÜRKSAT A.Ş. in Ankara.

...
...
...
...
...
...
...
...
...
...
...

...
...
...
...
...
...
...

7 References

https://bitbucket.org/temurtas/pi/
https://bitbucket.org/temurtas/staj_matlab
https://bitbucket.org/temurtas/ee300_report
https://pomotodo.com/app/
https://airtable.com/shrI9Y26ehXklCe9m
https://airtable.com/shrCJKhPqLuX9y0lh
https://guide.airtable.com/
https://www.coursera.org/learn/matlab
http://www.comparex-group.com/MediaLibrary/Catalog/web/topic/microsites/mslar/sharepoint_features_home.jpg