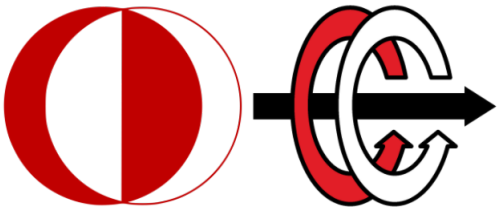

Middle East Technical University

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 **3**

2 Description of the Company **3**
2.1 Company Name 3
2.2 Company Location 4
2.3 General Description of the Company 4
2.4 The Organizational Chart of the Company 5
2.5 A Brief History of the Company 6

3 Orientation & Useful Programs **7**
3.1 Pomodoro Technique 7
3.1.1 Pomotodo App 8
3.2 Database Structure 10
3.2.1 Airtable 10
3.3 Wiki Pages 11
3.3.1 Confluence Wiki 12
3.4 V-Model & Agile Methodology 13
3.4.1 V-Model 13
3.4.2 Agile Methodology (Scrum) 15
3.4.2.1 Roles 15
3.5 Version Control with Git 15
3.5.1 Github 16
3.5.2 Bitbucket 16

4 Solar Tracker System Project **17**
4.1 Planning & Researching 17
4.1.1 System Requirements 19
4.1.2 Subsystem Requirements 19
4.1.3 Component Requirements 20
4.1.4 Components 20
4.2 Training 21
4.2.1 Training on Python 21
4.2.1.1 Basics 21
4.2.1.2 Using Conditions 23
4.2.1.3 Using Loops 23
4.2.1.4 Defining Functions 23

4.2.1.5 Defining Classes 24
4.3 Working on the Project 25
4.3.1 Working on Raspberry Pi 26
4.3.1.1 Training on LEDs & GPIOs 27
4.3.1.2 Training on LDRs 28
4.3.1.3 Training on Servo Motors 29
4.3.2 Raspberry Pi Final Code 30
4.3.3 Working on Arduino 30
4.3.3.1 Training on LEDs & Pins 31
4.3.3.2 Training on Servo Motors 32
4.3.4 Final Arduino Code 33
4.4 Implementation 33
4.4.1 PCB Drawing 33
4.4.2 Top Layer 34
4.4.3 Solar Panel 34
4.4.4 Main Body 35
4.4.5 Final Body 36
4.5 Tests 36
4.5.1 Component Requirement Tests 37
4.5.2 Subsystem Requirement Tests 38
4.5.3 System Requirement Tests 38

5 After Project **39**
5.1 Training on MATLAB 39
5.2 Training on Microsoft Sharepoint 40

6 Conclusion **41**

7 References **42**

A Raspberry Pi Final Code **43**

B Arduino Final Code **45**

C Simple Sorting Code in MATLAB **47**

1 Introduction

I 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 my supervisor and he managed my internship program.

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

The project was about solar panels system that can follow sun to increase its efficiency. In order to achieve this goal, we were recommended to use Raspberry Pi instead of Arduino since other team were using the 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 environment. 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 is 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.[11]

As of July 2017, Türksat A.Ş. has total number of 1122 employee working variety of departments. 129 of them being electronics engineer, 349 engineers works in the company. After electronics engineers, by 109 engineer computer science is second major group in working engineers. Directorate of Satellite Programming, the directorate I made my summer practice, has total number of 18 total employee with 15 engineers.

The work done at the department was to supervise and control the process and projects in the process of building the latest satellite. The engineers I work with were responsible in some parts of Türksat 6A project.

2.4 The Organizational Chart of the Company

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

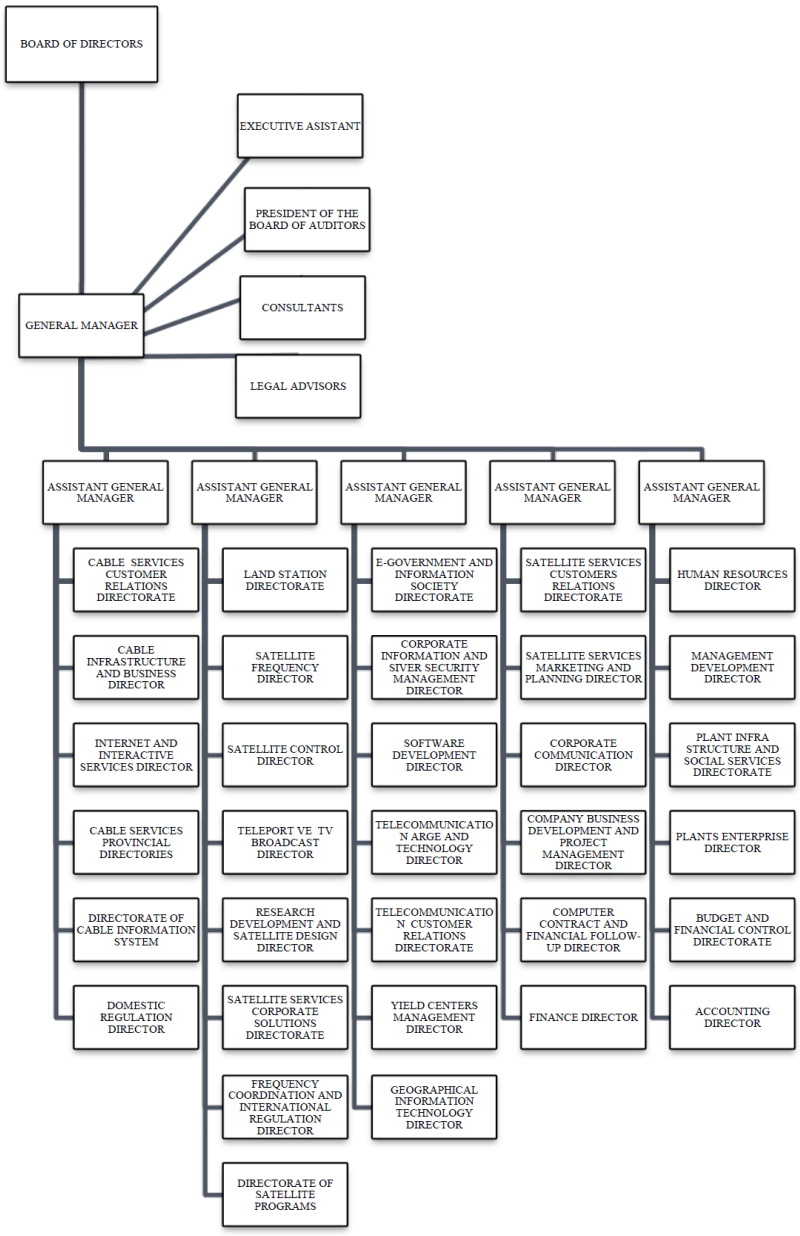

Figure 1: The Organizational Chart of TÜRKSAT

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 my supervisor.

In this section, I will explain the techniques and programs that I found quite 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 length of this intervals can be changed by people's personalities. For example, I have used 40 minutes lenght pomodoros, 5 minutes length short breaks and 1 hour length 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. [10]

The technique is closely related to software design concepts such as incremental development and iterative and time-boxing, 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 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. I used more technological solutions called Pomotodo App in my summer practice.

The reason behind this decision was to increase efficiency even more 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.[6] 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 categories. Some of my to-do list objects can be seen at *Figure 2*

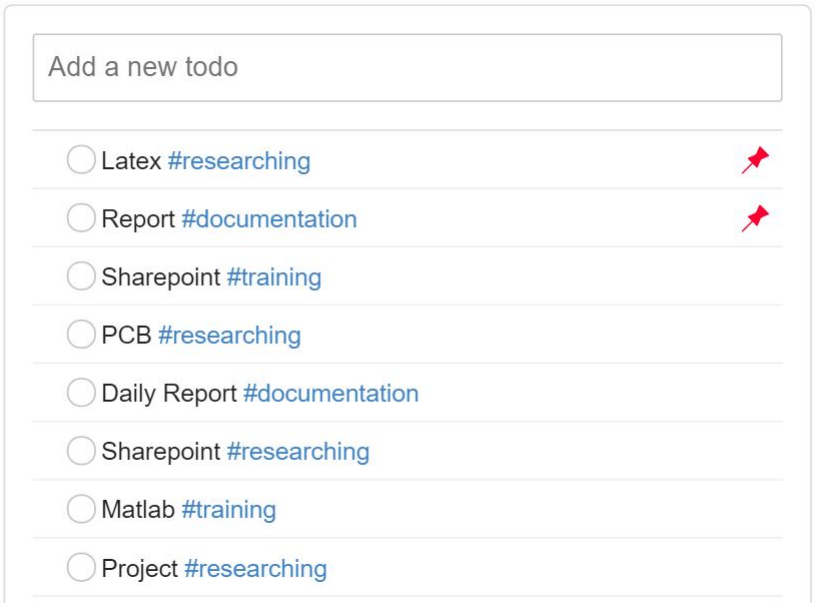

Figure 2: My To Do List in Pomotodo Web App

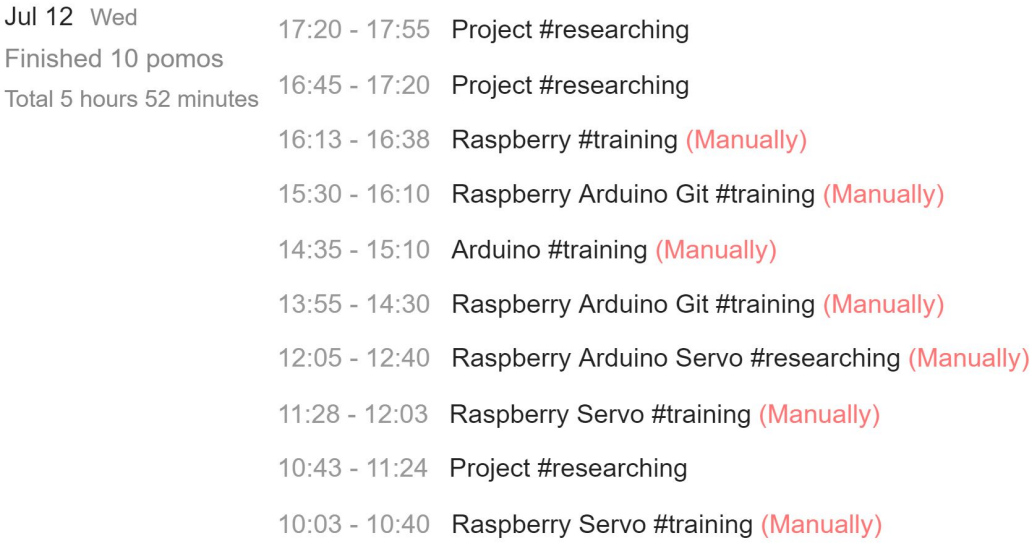

Figure 3: My Pomodoro History of July 12th

As can be seen at *Figure 3*, ten pomodoros were completed at July 12th and some hashtags were used to categorise the work done. As I mentioned earlier, I have tried to use my pomodoro length as a 40 minutes and short breaks as 5 minutes. After five completed pomodoros, a long break was taken. Thanks to this hashtag usage, I can investigate 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*.

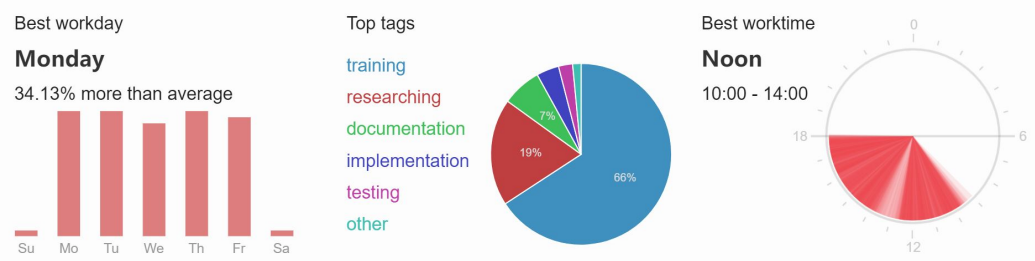

Figure 4: Some statics about my summer practice

3.2 Database Structure

A database can be defined as 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 real life situations in a way that supports processes requiring information, such as modelling the necessity of component in project in a way that finding the most suitable parts for the project in budget would be easier.

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. 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.

Due to their close relationship, the term "database" is often used to refer to both a database and the DBMS used to manipulate it. Outside the professional information technology world, database term is often used to refer to any collection of related data such as a spreadsheet or a card index.

In my summer practice, I have used Airtable which is a unpaid spreadsheet-database hybrid solution for storing the necessary data and manipulating it.

3.2.1 Airtable

Being a spreadsheet-database hybrid, Airtable applies the features of a database 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. Thanks to its variety of abilities, I could create a database, set up field types, add records, link tables, collaborate with my team mates, sort the records based on a field. Since it is automatically hosted to the cloud, the values in the fields are updated real time. [9].

Airtable has five basic components

1. **Bases** : All the information needed to create a project is contained in a base. It includes subsections like tables, views and so on. Bases me and my team mates have used for our project can be seen at *Figure 5*. Moreover, a sample Apartment Hunting base can be seen at *Figure 6*.

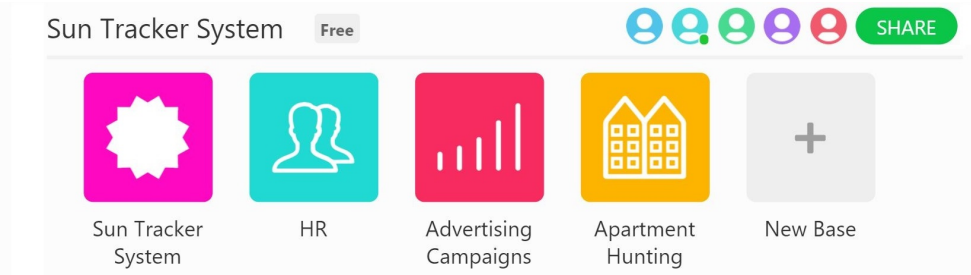

Figure 5: Bases I have used for Sun Tracker System

2. **Tables** : A table is similar to an excel spreadsheet. A Base is a collection of tables. Apartments & Districts tables that form Apartment Hunting Base can also be seen at *Figure 6*.

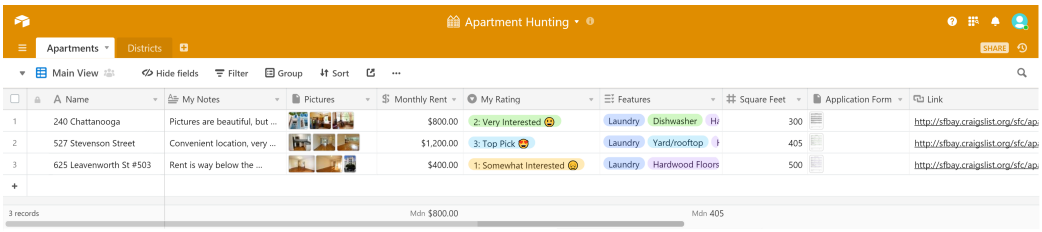

Figure 6: Apartment Hunting Base

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

4. **Fields** : Each entry in a Table is a field. They can be thought as columns in Excel spreadsheet. Unlike Excel, Airtable offers 16 basic field types like single-line texts, file attachments and so on.

5. **Records** : Each row of a Table is a Record. They can be thought as rows in Excel spreadsheet.

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.[4]

A wiki is run using wiki software, otherwise known as a wiki engine. A wiki engine is a type of content management system, but unlike most other such systems, including blog software, the content is created without any defined owner or leader. 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. 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.

In my summer practice, I have used Confluence Wiki on my supervisor's advice. Thanks to its abilities, we could share knowledge with other interns even though we were not on same project team.

3.3.1 Confluence Wiki

Being a product of Atlassian, Confuluence is one of the most used wiki engines in professional world. Including NASA and Türksat, where I made my summer practice in, over 35,000 company or team uses Confluence Wiki in their projects. Main Interface of my wiki page in Confluence Wiki can be seen at *Figure 7*.

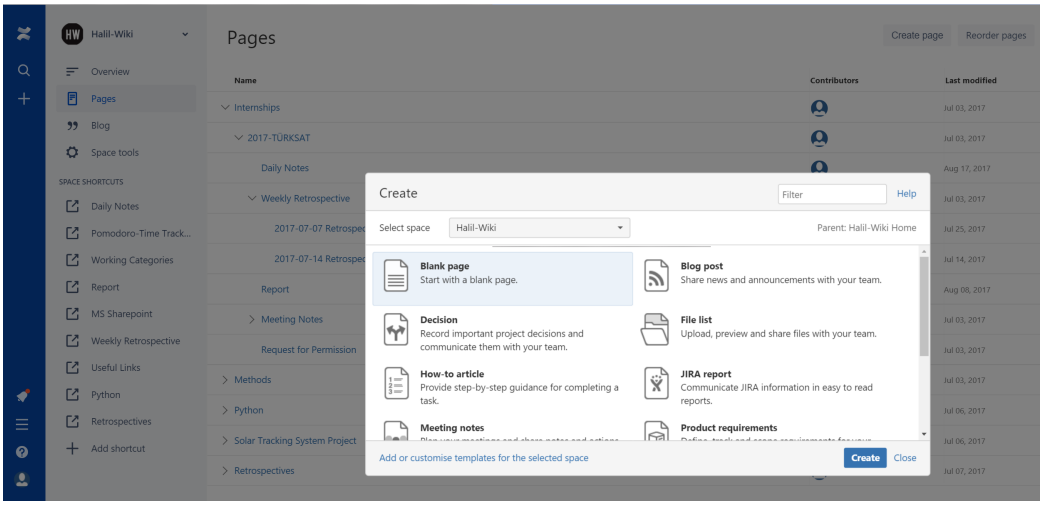

Figure 7: Confluence Wiki

I have used software mostly for keeping daily notes, tracking my work on

project and sharing knowledge with other interns. For instance, I created a page inside my wiki-page called **Daily Notes** in order to track my work at Türksat. In that page, I took daily notes about the project I have been working on and other works done each day. Moreover to decide what I will do in rest of my summer practice, I have crated to-do lists assigned to me with a due date in a same page. As can be seen at can be seen at *Figure 8*, I have also included my **Pomotodo Daily Reports** to the same page in order to keep track the time spend on each task.

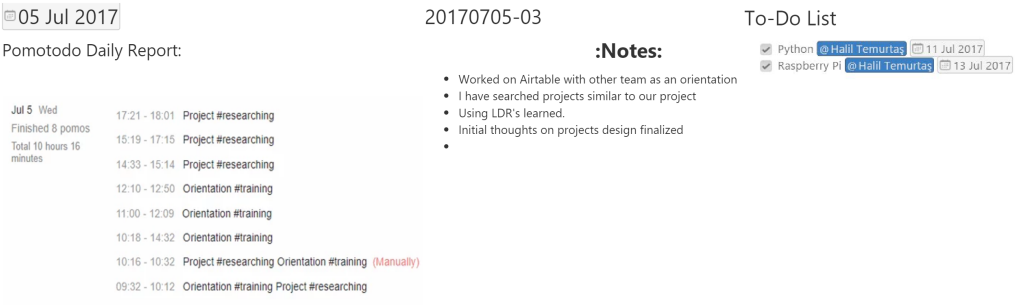

Figure 8: My Notes from July 5th in Daily Notes Page

3.4 V-Model & Agile Methodology

Throughout my summer practice, several methodologies were applied in order to increase efficiency. V-model and Agile methodology were strongly recommended by my supervisor. Since these techniques are used widely by major companies in the industry, it would be very beneficial for my later work life.

In this section, I will explain the versions of V-model and Agile methodology I used in my internship.

3.4.1 V-Model

The V-model is a graphical representation of a systems development lifecycle. The system is used to produce extremely accurate development lifecycle models and project management models. The V-model consists of three major categories, the requirements, design&implementation and tests. The basic model I used can be seen at *Figure 9*.

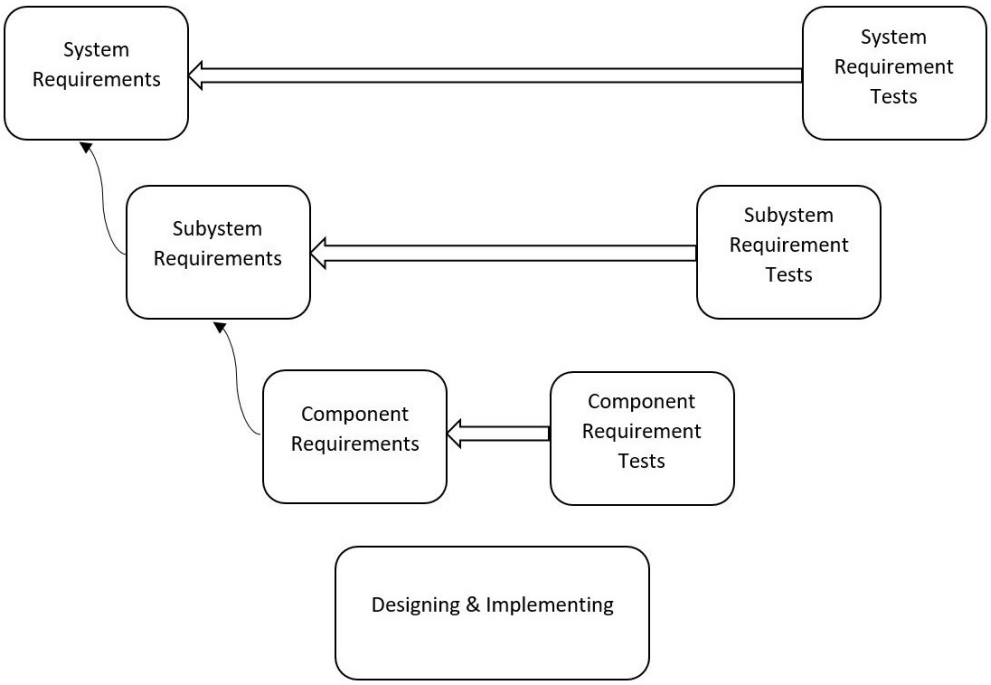

Figure 9: V-Model Schematic

The left side of the **"V"** represents the main requirements, and creation of system specifications while the right side of the **"V"** represents validation of parts and their tests. 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. The easiest way is to say that verification is always against the requirements and validation always against the real world or the user needs.

In our project, for creating **"V"**, we firstly specify the requirements of the project. To do that effectively, system requirements were constructed. Under these requiremets, subsystem requirements were decided. Moreover, in order to chose right components for the project, component requirements were decided by considering subsystem requirements. Finally, the test were formed by considering the each requirements. For the simplicity, the required V-model's constructed using Airtable. The base created for the **Sun Tracker System** can be examined in Airtable [7].

3.4.2 Agile Methodology (Scrum)

Mostly used for software development, Agile describes a set of values and principles for project development under which requirements and solutions evolve through the collaborative effort of teams. It advocates adaptive planning, evolutionary development, early delivery, and continuous improvement. Moreover, it encourages rapid and flexible response to change.

The term 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 used Scrum for our project. As with many methodologies, Scrum also has roles for sharing the work within a team. Thanks to our supervisor, we decided to use six different role. Scrum master being our supervisor, we split these roles between me and my team-mates. These collaboration will be explained deeply in the project section.

3.4.2.1 Roles

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.

Unlike most client–server systems and as with most other distributed version control systems, every Git directory on every computer is a full-fledged repository with complete history and full version tracking abilities. Thus, git is independent of network access or a central server.

Throughout my summer practice, I tried to use these system for controlling my coding attempts. Starting on my Windows laptop, I have used git mainly on Raspberry Pi and Linux environment.

3.5.1 Github

GitHub is a web-based Git or version control repository and Internet hosting service. Mostly used for code, it offers all of the distributed version control and source code management functionality of Git as well as adding its own features. It provides access control and several collaboration features such as bug tracking, feature requests, task management, and wikis for every project.

I used Github while learning the basics of git and version control. After first few days, I started to use Bitbucket for personal preferences.

3.5.2 Bitbucket

Bitbucket is another 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 integrates with other Atlassian software like Jira, HipChat, **Confluence** and Bamboo. Bitbucket has traditionally tailored itself towards helping professional developers with private proprietary code. The interface of Bitbucket and source code comparison in Bitbucket can be seen at *Figure 10.*[2]

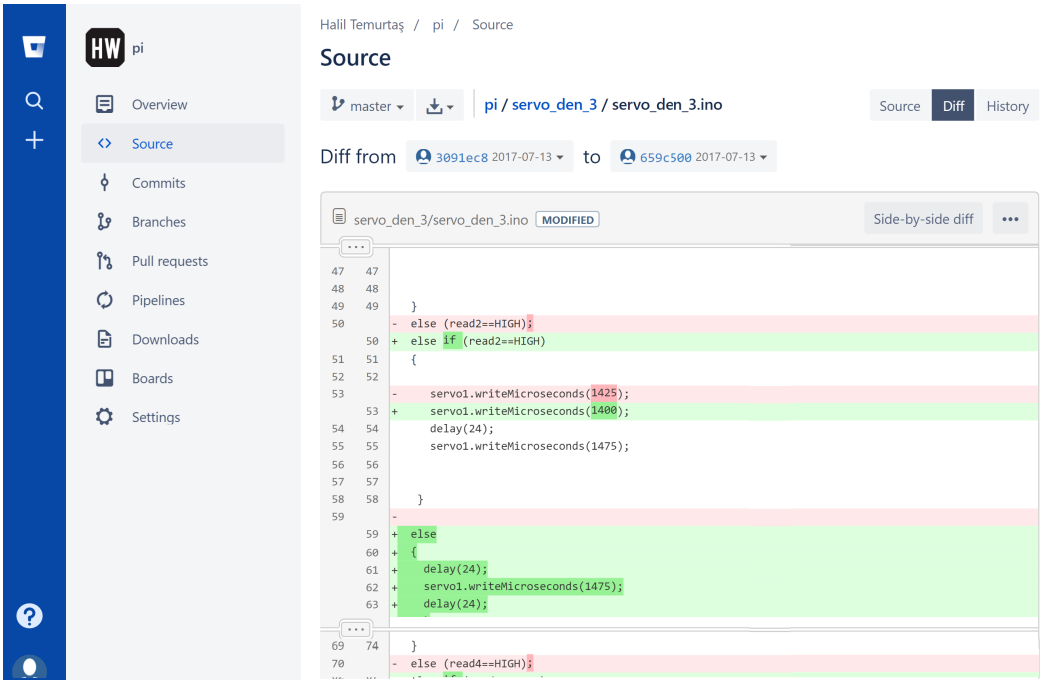

Figure 10: Version Control with Bitbucket

4 Solar Tracker System Project

In my summer practice, I was assigned for a project with a team. For the project, I was 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, I 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. Thus, we decided to use Airtable for tracking system requirements, tests and so on. The Interface of Airtable & System Requirements can be seen at *Figure 12*.

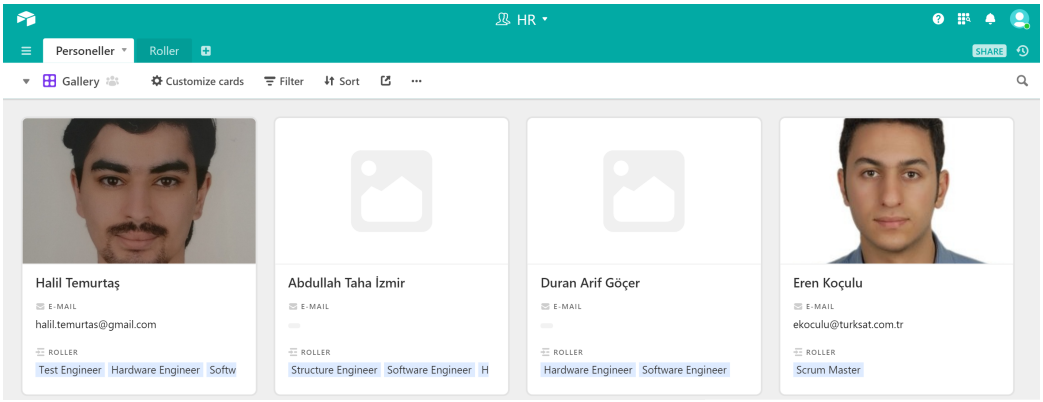

Figure 11: HR Base in the Airtabel

While planning the project, some roles were assigned within our team that can be seen via our HR base in Airtable [8] in *Figure 11.* 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 1.* The responsibilities of these roles were explained in Section 3.4.2.1. in *page 15.*

	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 1: Roles

While researching, I looked for similar projects that uses a micro-controller or embedded systems for tracing space objects like Sun, ISS, moon and so on. Analysing these projects[13, 14] helped me finalize initial thoughts about the project and create needed requirements for it.

4.1.1 System Requirements

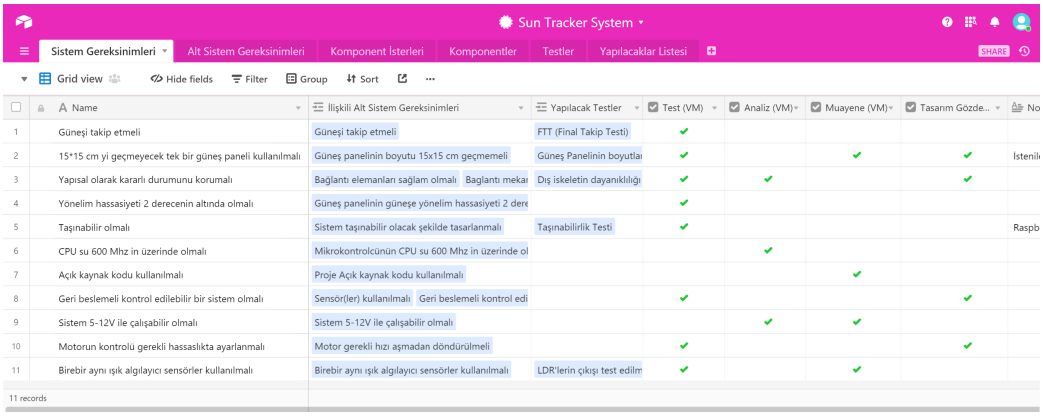

Figure 12: The Interface of Airtable & System Requirements[7]

Constructing V-model, see *Figure 9*, required us 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 the project and it became one system requirement. 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 12*.

4.1.2 Subsystem Requirements

As mentioned 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. At the end of this process, subsystem requirements would be decided. As can be seen at *Figure 13*, we had 14 subsystem requirements for finalizing the project.

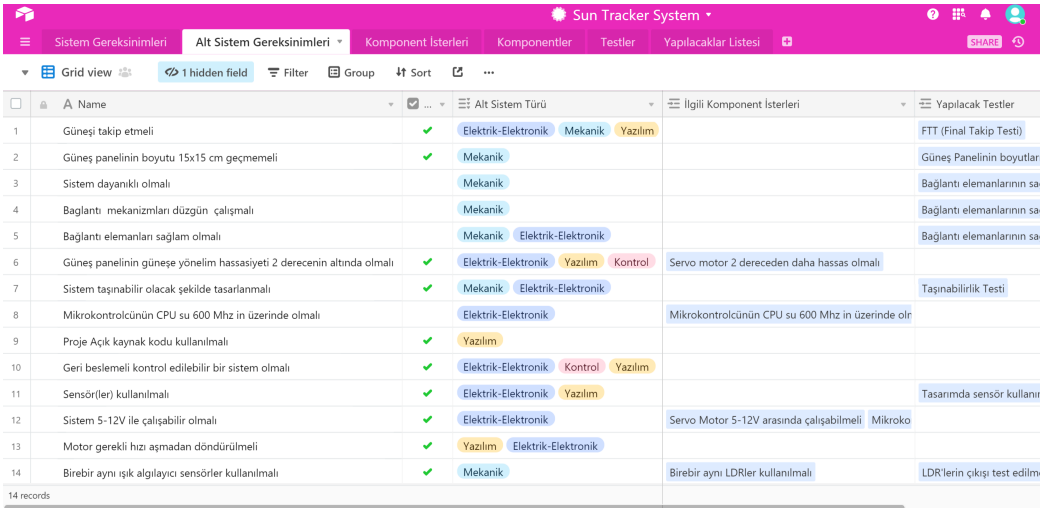

Figure 13: The Interface of Airtable & Subsystem Requirements[7]

4.1.3 Component Requirements

Choosing the right components can be one of the most challenging parts of the projects and it is vital for success of final results. V-model suggest very good solution for handling this problem by its nature. By considering the subsystem requirements, priorities the components must be fulfilled are considered. By the end of this process, component requirements would be decided. For our project, we decided six component requirements as can be seen at *Figure 14*.

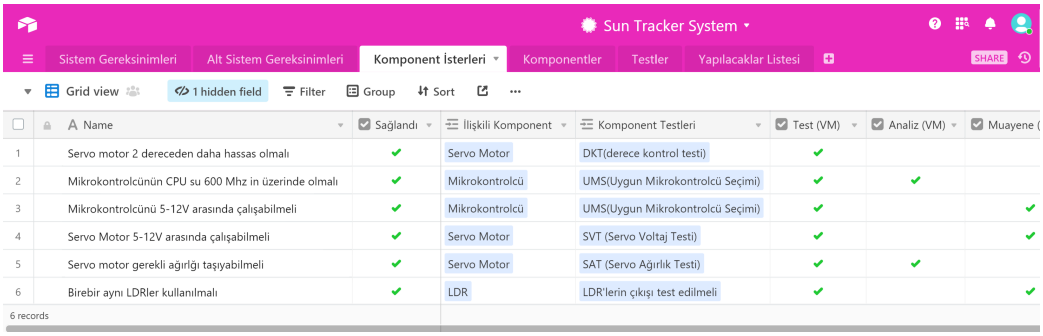

Figure 14: The Interface of Airtable & Component Requirements[7]

4.1.4 Components

After deciding on the component requirements, needed component were chosen according to these requirements. Necessary components were ordered

online. As can be seen at *Table 2*, two micro-controller were used for the project. Raspberry Pi 3 was used as a main micro-controller and Arduino Uno was used as a secondary micro-controller.

Table 2

	Number used	Unit Price (TL/unit)	Cost(TL)
Servo Motor	2	53.74	107.48
Microcontroller	1	158.9	158.90
Additional microcontroller	1	50.00	50.00
Solar Panel	2	23.18	46.36
LDR	4	1.26	5.04
Jumper	80	0.12	9.60
Hex Nut	10	0.01	0.14
USB Voltage Regulator	1	7.21	7.21
Battery Holder	1	1.60	1.60
Screw	20	0.07	1.34
Standoff	16	0.19	3.04
Heat Shrink (1 meter)	1	1.26	1.26
Outer Material	2	15.00	30.00
TOTAL			421.97

Table 2: The components used in the project.

4.2 Training

Throughout my summer practice, I spent most of my time on training, see *Figure 4* on page 9. In this section, I will explain what I trained in throughout the project.

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.

4.2.1.1 Basics

Here are some of my very first attempts to use Python.

```python
# 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)

print(numbers)  # prints [1, 2]

names = ["Ali", "Ahmet", "Ayse"]  # adds Ali, Ahmet and Ayse
                                  to names.
second_name=names[1]
print("The 2nd name on the name list is %s" %second_name)
                                     # prints 'The 2nd name on
                                  the names list is Ahmet!'

astring = "Hello world!"

print(astring.index("o")) # prints 4, since o appears firstly
                                   at 4\nth digit.
print(astring.count("l")) # prints 3, since l appears three
                                  times
print(astring[3:7])   # prints lo w, starting from 3rd
                                  element to 7\nth element (7\
                                  nth is not included!)
print(astring[3:7:2]) # prints l, starting from 3rd element
                     #to 7\nth 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

As with many other programming languages, using conditions is very essential part of programming in Python. Owing to that reason, learning to use them was priority for both learning Python and project. The basic structure for using conditions can be seen below.

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

4.2.1.3 Using Loops

Like using conditions, using loops is inseparable part of programming in Python. The basic structure for using 'for' and 'while' can be seen below.

Using For Loop

```
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]
```

Using While Loop

```
count=0
while (count<5) :
    print(count)  # prints the count in every loop
    count +=1   # by adding one each time
else:
    print("count value reached %d" %(count))
```

4.2.1.4 Defining Functions

While developing a software program, building an algorithm as one executable program or function can be very complicated. Defining different

functions for each task can save a lot of time and prevent misunderstandings in the process. The basic structure for defining new functions in Python can be observed below with an easy example.

```
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 thanks to defined
                              function!
print("x=%s" %x)  #prints 'x=3'
```

4.2.1.5 Defining Classes

Defining classes can help developing better software due to similar reasons with defining function. The basic structure for how to define new classes with an easy example can be seen below.

```
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, I realised that Python is not very difficult language to learn. In fact, aside from some indent mistakes, using Python as a programming language is very simple and clean yet powerful in various applications.

4.3 Working on the Project

Event though I wanted to separate this section from training section, I spesified the work done in this section as '#training' in my daily pomos. Thus, the work spent on training seems higher than usual, see *Figure 4* on page 9. Throughout the project, I tried to use git for code work. My commit history on Raspberry Pi can be seen st *Figure 16*,

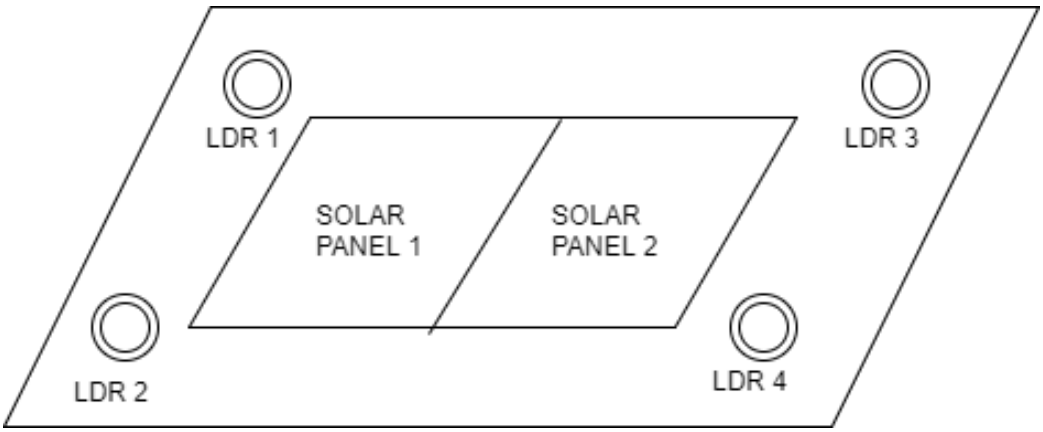

Figure 15: Main Draft for the Top Panel of the Sun Tracker

The goal in this project was to design a solar panel system that can follow sun light in order to increase its efficiency. To do that, we decided to use four LDRs at the four corners for solar panels as can be seen from *Figure 15*. Then, using the data coming from these sensors the platform would return to the side where the sun light would be maximum. One of the servos was responsible for vertical movement and other servo was responsible for horizontal movement in the project design. Although we planned to use Raspberry Pi as an only micro-controller. The difficulties forced us to use Arduino as an supportive micro-controller in the project.

In this section, I will explain the work done for the project step by step.

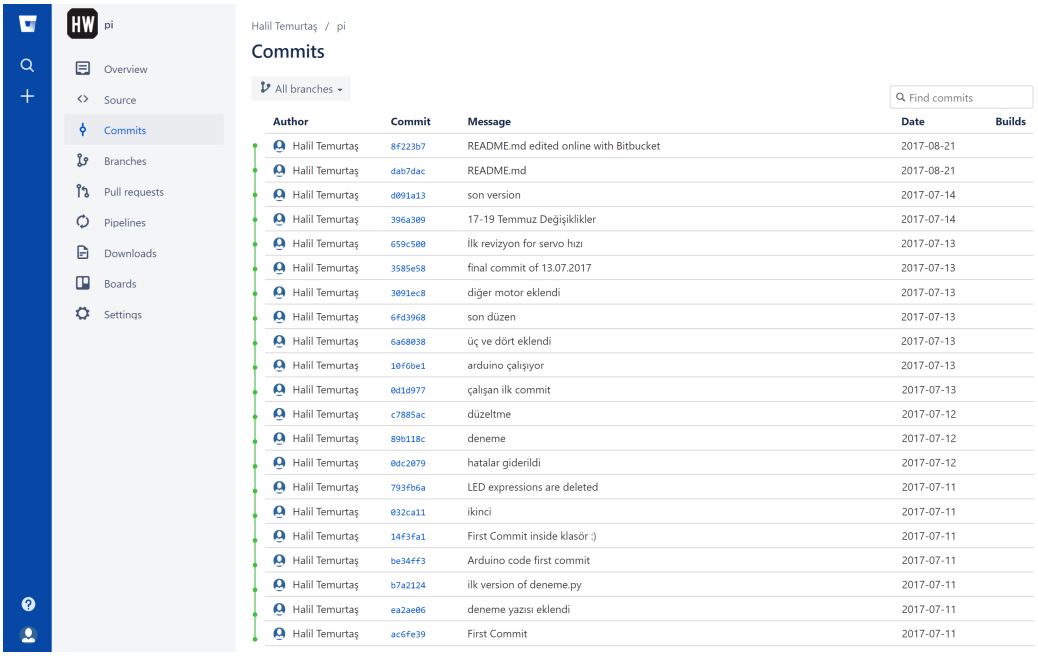

Figure 16: The Interface of Bitbucket & My Commit History

4.3.1 Working on Raspberry Pi

Since we wanted to use Raspberry Pi as an main micro-controller in our project, we moved on working on that immediately after learning the basics of the Python. Very first line of codes, I wrote on Raspberry can be seen below.

```
1 x==4   # first line of code on raspberry pi
2 if x==4    # no ';' or ':'
3     print("evet")    # My version of 'Hello World'!
```

As can be seen from *Figure 17*, Raspberry Pi 3 has 40 pins ,in other words, GPIOs (General-Purpose Input/Output). Numbers in the grey boxes can be used for setting desired GPIO as input or output by proper code. These numbers are defined by Broadcom SOC channel and universal.

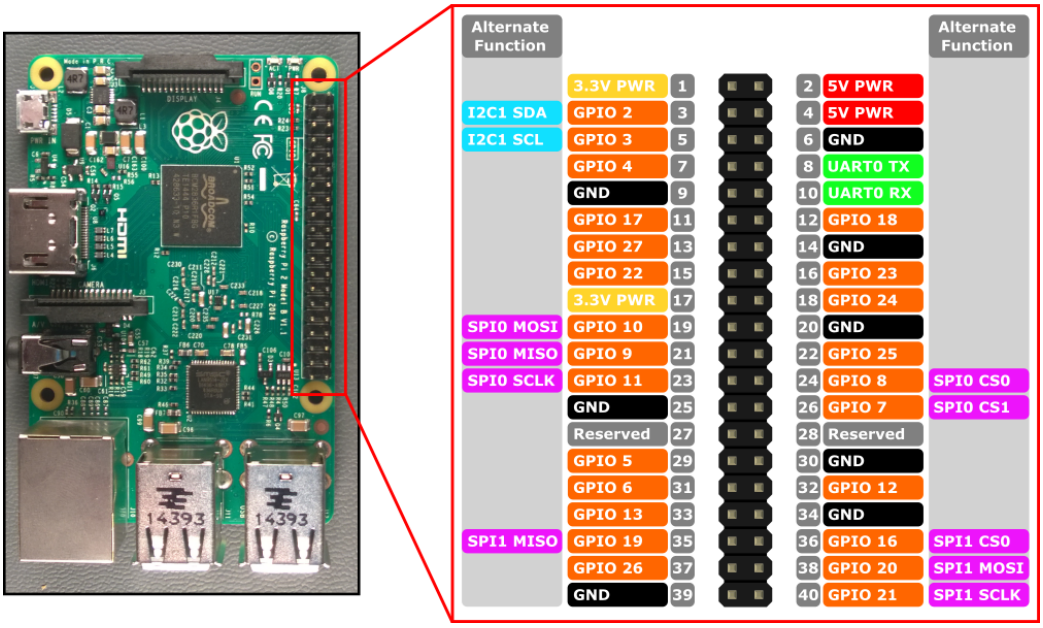

Figure 17: Raspberry Pi 3 & Pin Mapping[1]

4.3.1.1 Training on LEDs & GPIOs

In order to use Raspberry, learning the usage of GPIOs is very crucial. Therefore, first thing I did was to try them using LEDs since visual outputs of LEDs can inform us about process. The simple algorithm for blinking LEDs can be seen below.

```
1  import RPi.GPIO as GPIO    #imports necessary library.
2  import time
3
4  GPIO.setmode(GPIO.BCM)  #calls GPIOs by numbers defined by
                                Broadcom SOC channel
5  GPIO.setwarnings(False)
6  GPIO.setup(17,GPIO.OUT) #sets GPIO 17 as an output pin
7  GPIO.setup(4,GPIO.OUT)  #sest GPIO 4 as an output pin
8
9  while True:   #sets an infinite loop
10     print "LED on"
11     GPIO.output(17,GPIO.HIGH)
12     GPIO.output(4,GPIO.LOW)
13     time.sleep(1)   #sets sleep time
14
```

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

4.3.1.2 Training on LDRs

As planning the project, we decided that the cheapest yet effective way to track light around the panels would be using LDRs. Thus, after gaining some experience on using GPIOs, I spent some time on learning the usage of LDRs with Raspberry.

Since Raspberry Pi does not have analog GPIOs and LDRs are analog sensors, I needed to find a way to communicate with it through digital GPIOs. After some research, I found a library that uses time takes to reach 1.3 volts at the output of LDR, since after 1.3 volts the input GPIO turns 'on' rather than 'off'. Here is the basic algorithm for using four LDRs at the same time.

```
from gpiozero import LightSensor, Buzzer  #imports library

ldr = LightSensor(4)  #Assigns the GPIO 4 to LDR1
ldr2 = LightSensor(17) #Assigns the GPIO 17 to LDR2
ldr3 = LightSensor(27) #Assigns the GPIO 27 to LDR3
ldr4 = LightSensor(22) #Assigns the GPIO 22 to LDR4

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

After spending some time on LDRs, I moved on servo motors. A servo-motor is a special kind of motor that allows for precise control of angular or linear position, velocity and acceleration. After some research and dozens of different algorithm trial, I could not write a proper algorithm for controlling servo we bought. One of these algorithm can be seen below.

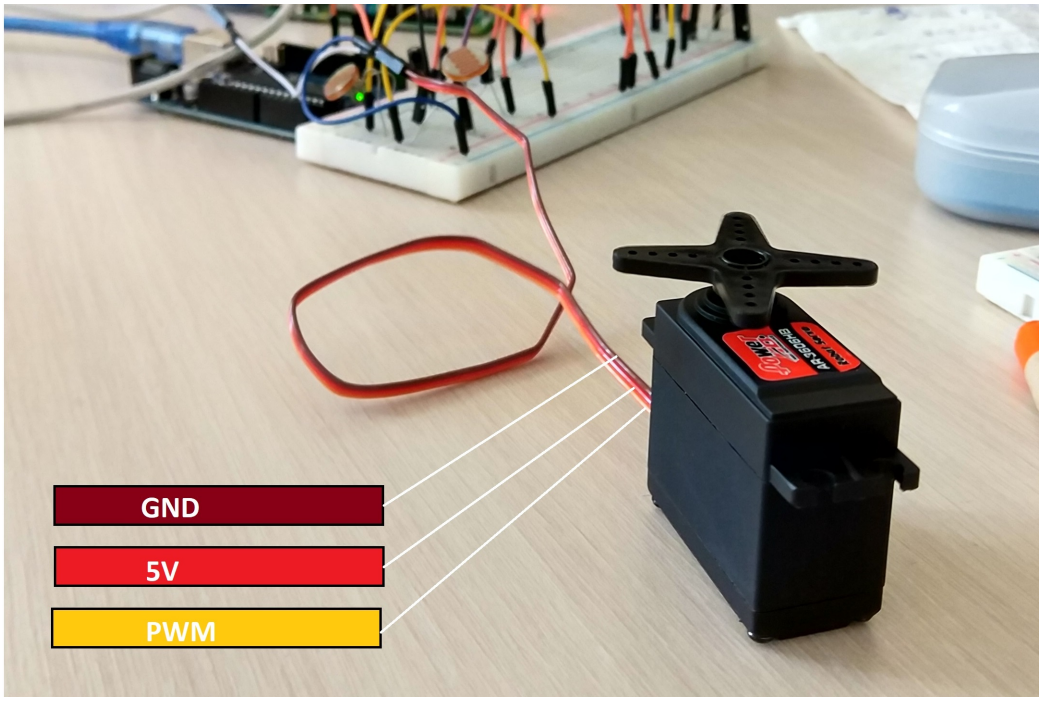

Figure 18: One of servo motors used and its inputs

```
# 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)
```

```
10 delay_period = 0.01
11
12 while True:
13     for pulse in range(50, 250, 1):
14         wiringpi.pwmWrite(18, pulse)
15         time.sleep(delay\_period)
16     for pulse in range(250, 50, -1):
17         wiringpi.pwmWrite(18, pulse)
18         time.sleep(delay_period)
```

Realising that our servo motor actually a continues servo motor that can turn 360 degree without an error and be controlled by rotation speed. The efforts to control servo failed since all of libraries I found worked with defining angle rather than speed. And in this process, we learnt that Raspberry Pi do not have enough PWM output to supply two servos and we needed to buy a motor controller. Thus, we decided to use Arduino to control servo and communicate between two micro-controller via jumpers. The continuous servo we bought and its inputs can be seen at *Figure 18*, like many other regular servo it has tree inputs.

4.3.2 Raspberry Pi Final Code

Raspberry Pi part of final algorithm for Sun Tracker System can be found at Appendix A or inside the Bitbucket repository for the project[2].

4.3.3 Working on Arduino

After spending couple of days on driving servo motors with Raspberry Pi, I decided it would be much easier to use Arduino for that work. Our plan was to connect through jumpers so that we did not have to use any serial connection or code based communication between two micro-controller. Since our project was not too complicated for that purpose, we were able to achieve what we wanted to do at the beginning.

Four jumpers were used to communicate two devices. Thanks to algorithm used in the Raspberry Pi, whenever the servo is required to driven, one of the output GPIOs became high and otherwise became low. Then, connected pins at the Arduino side are checked with 'digitalRead' function. With a necessary algorithm, see Appendix B, the pins used for PWM outputs for servos controlled with the help of this readings. The pin connection between Arduino Uno & Raspberry Pi can be seen at *Figure 19*.

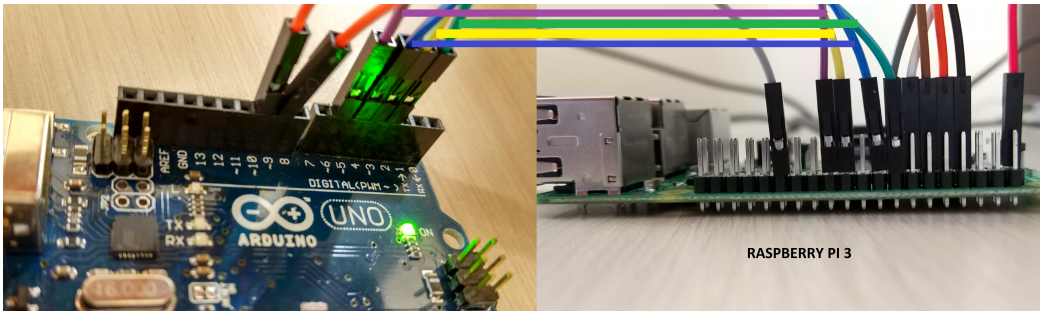

Figure 19: The Connection Between Arduino Uno & Raspberry Pi

4.3.3.1 Training on LEDs & Pins

In order to use Arduino, learning the usage of pins is very essential like using the GPIOs in the Raspberry Pi. Thus, first thing I did was to try them using LEDs since outputs can be observed very easily thanks to visual outputs of LEDs. The simple algorithm for two blinking LEDs using the 'digitalWrite' function can be seen below.

```
void setup() {  // the setup function runs once at the beginning
    pinMode(9, OUTPUT); //sets pin 9 as an output pin
    pinMode(15, OUTPUT);  //sets pin 15 as an output pin
}

void loop() { // the loop function runs forever
  digitalWrite(9, HIGH);    // turns the LED connected to pin 9
                               on (HIGH is the voltage level)
  digitalWrite(15, LOW);
  delay(1000);                // waits for a second
  digitalWrite(15, HIGH);
  digitalWrite(9, LOW);     // turns the LED off by making the
                               voltage LOW
  delay(1000);                // waits for a second
}
```

4.3.3.2 Training on Servo Motors

In order to finish the project with a success, servo motors should be driven according to data coming from LDRs. The first part was done on the Raspberry Pi so only thing left was to figure out how we can drive servo motors using Arduino. Fortunately, there is a very useful library for controlling servos called 'Servo.h'. Firstly, we tried to use servo.write function as in the code below. Since this function was originally designed for position control and its rage was from 0 to 180, we decided to use **servo.writeMicroseconds** function instead since its range was far wider than servo.write function.

With ideal continues servos servo.writeMicroseconds(1500) command stops the motor. However, I found out that our servo actually stop with servo.writeMicroseconds(1475). As the number departs from 1475, the servo attached the function turns with a higher speed in opposite direction[2].

My very first attempts to drive servos using 'servo.write' function can be seen below.

```
#include <Servo.h>

Servo Servo1; //create servo named Servo1 to control first servo
Servo Servo2; //create servo named Servo2 to control other servo

void setup()
{
  Servo1.attach(9);  //attaches the servo on pin 9 to the Servo1
  Servo2.attach(10);//attaches the servo on pin 10 to the Servo2
}

void loop()
{
    Servo1.write(100);
    Servo2.write(90); // tell Servo2 to stop
    delay(15);  // waits 15ms
    Servo1.write(90); // tell Servo1 to stop
    Servo2.write(80);
    delay(15);  // waits 15ms

}
```

4.3.4 Final Arduino Code

Arduino Uno part of final algorithm for Sun Tracker System can be found at Appendix B or inside the servo_den_3 folder in the Bitbucket repository for the project[2].

4.4 Implementation

After completing the circuit and algorithm working on the breadboards, we moved on constructing body. For that purpose, we used hard plastic material for body. In this process we worked in the workshop instead of office.

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 20* and the final result that we used soldiring can be seen at *Figure 21*.

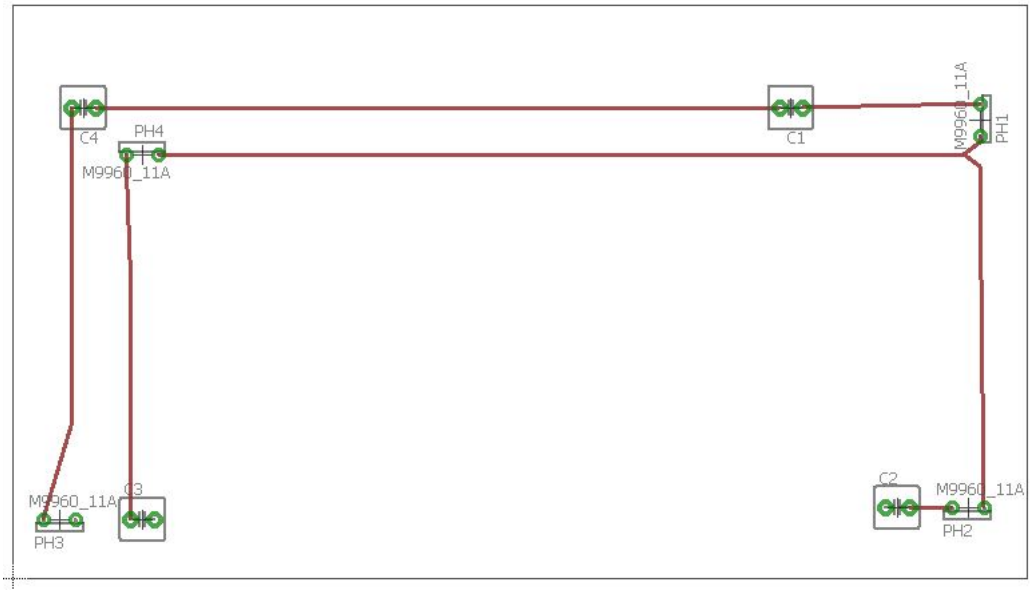

Figure 20: First attempts to draw PCB

4.4.2 Top Layer

Considering the design ideas from *Figure 15*, we cut a square shaped plastic plate that carries LDRs and solar panels. The connections between sensors, voltage supplies and grounds were formed by soldering jumpers.

To use outcome voltage of solar panel, we decided to implement a voltage regulator that accepts various input voltages and gave 5V at the output as an USB. Our aim here was to charge our phones with the system. Unfortunately, even though the connections were right and there was enough voltage at the output of the panels, we was not able to charge our phones using USB.

The final version of the top layer can be seen at *Figure 21*

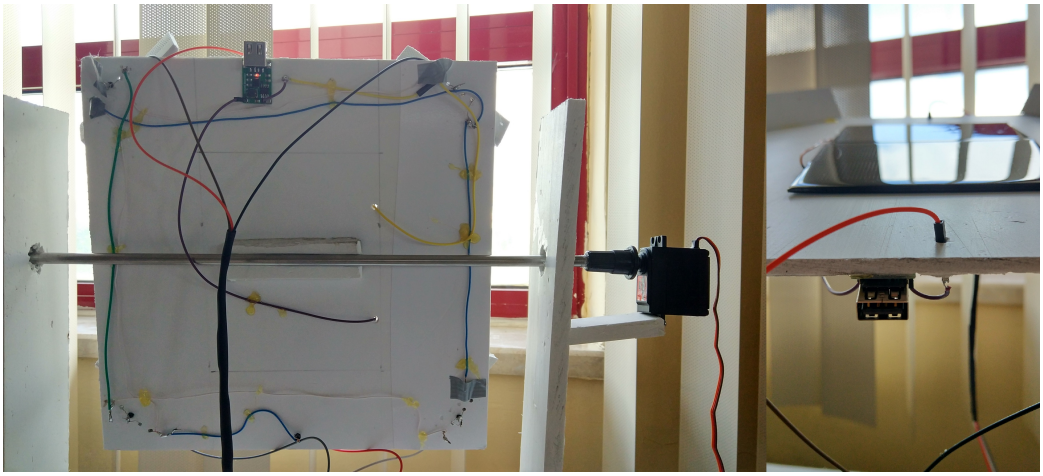

Figure 21: Top Layer

4.4.3 Solar Panel

As can be understood from the project name, our project was all about solar panels. 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 would be used in order to charge the phones and other USB powered devices. 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 21* and panels can be seen at *Figure 21* and *Figure 22*.

4.4.4 Main Body

While constructing the main body, we tried to build a dual axis tracker that allows panel to rotate in three dimensions. Due to fact that our servos can carry up to three kilograms, we mount the upper body directly to the first servo. In order to ease the rotation of top layer, two bearings were added to upper body.

The final version of the main body can be seen at *Figure 22*

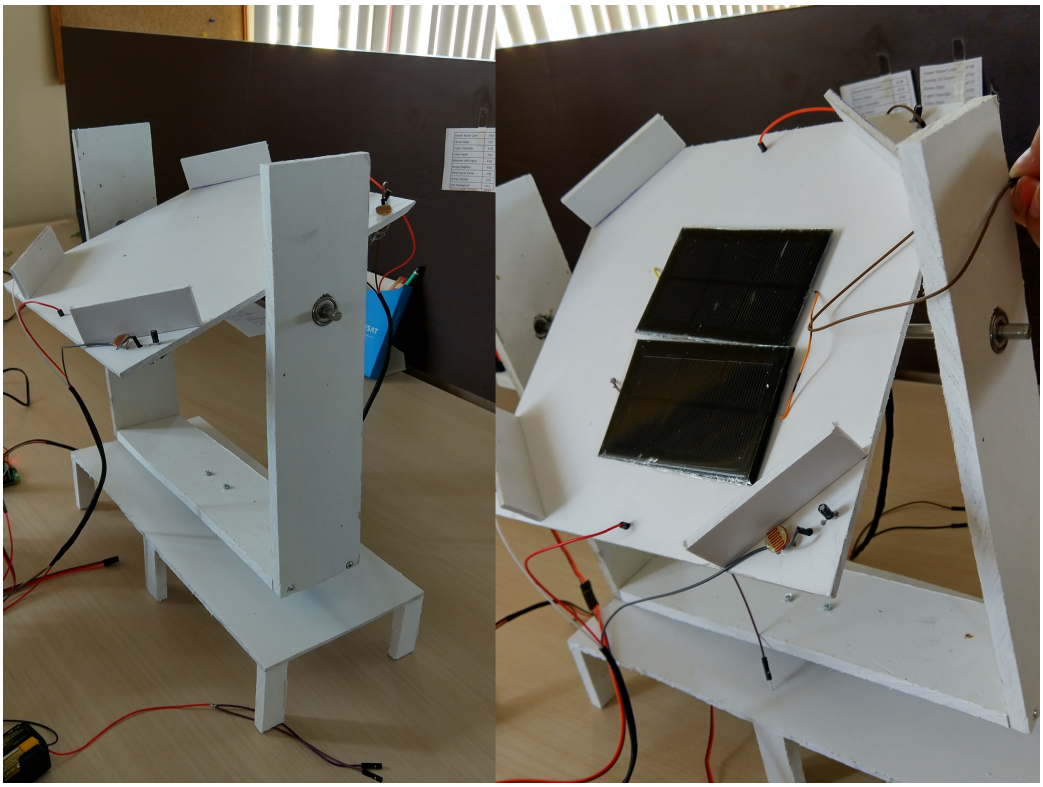

Figure 22: Body

4.4.5 Final Body

After a couple of days and lots of hand work, we constructed the final body that can be seen at *Figure 23*. Then, we moved on the tests which we should fulfil in order to complete the V-Model. By doing so, we made necessary adjustments to create final versions of algorithms.[**A**][**B**]

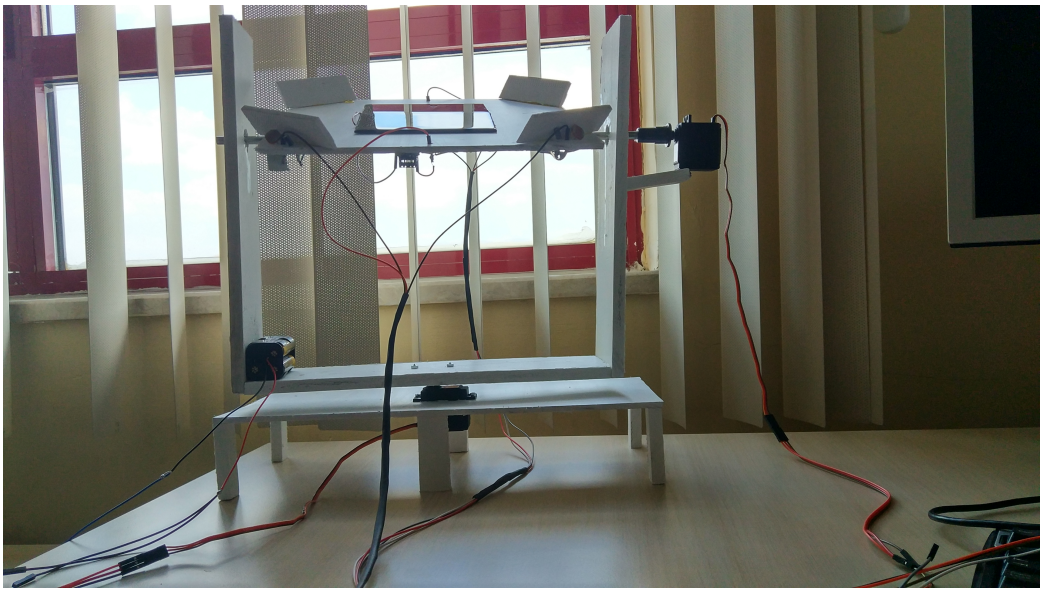

Figure 23: Final Body

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 subsystem tests. Therefore, most of the system requirement tests was not completed. Whole tests and the ones accomplished can be seen at *Figure 24*.

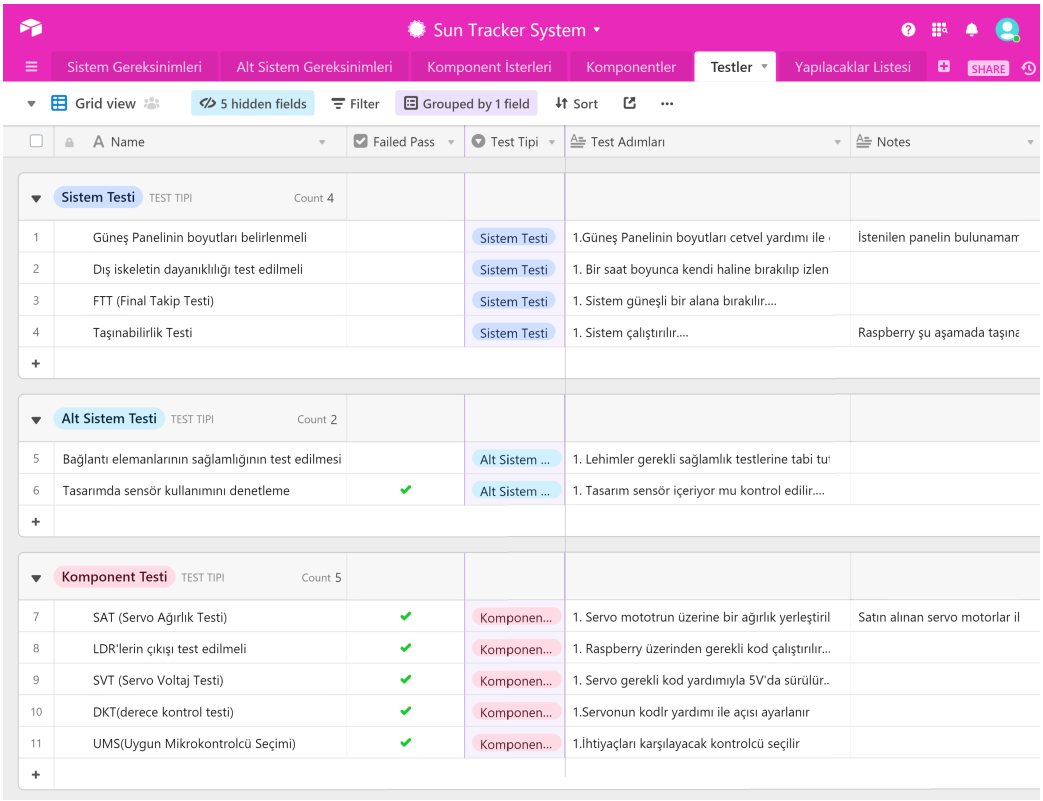

Figure 24: The Interface of Airtable & Tests

4.5.1 Component Requirement Tests

As V-Model suggests, we started with the component tests which decides whether the specific component requirement is completed. Thanks to right component choises, the project succesfully passed whole component requirements. All component tests can be seen at *Figure 24*. Whole test steps can be examined at Airtable[7]. Since the subsystem requirements are connected to the component tests, we moved on to subsystem tests.

4.5.2 Subsystem Requirement Tests

As V-Model requires, we moved on to subsystem tests after completing the necessary component tests. Due to lack of durability of soldered pieces and handmade connections, our project could not pass all subsystem tests. Whole test steps can be examined at Airtable[7]. Since the system requirements are connected to the subsystem tests, we could not move on to system tests.

4.5.3 System Requirement Tests

Due to problems at the subsystem tests, we were not able to pass system test as V-Model requires. Even though we passed this part, whole test steps can still be examined at Airtable[7].

My teammates, Taha & Arif, can be seen at *Figure 25*, while trying to stabilise the system in front of the real sized version of sun tracker at the garden of Gazi Teknopark as I took photos.

Figure 25: Real life tests in front of a real Sun Tracker

5 After Project

After finishing the project earlier than expected, I was asked to study for educational purposes since there was not enough time to finish another project. 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 and MS Sharepoint to practice on with my supervisor.

5.1 Training on MATLAB

For that purpose, I have enrolled a course on Coursera. Coursera, founded in 2012 by two Stanford Computer Science professors, offers over 2000 courses from 149 universities to almost 25 million users[12].

In the first two weeks of the course program, Matlab environment and basic operators were introduced. Since I know them already, I have watched these video lectures in a few hours. After that, I continued to rest of the course that aims to help users to learn the shortcuts and useful things about Matlab. The contents for second three weeks of course can be seen *Figure 26*. Rest of the course can be examined at coursera[12].

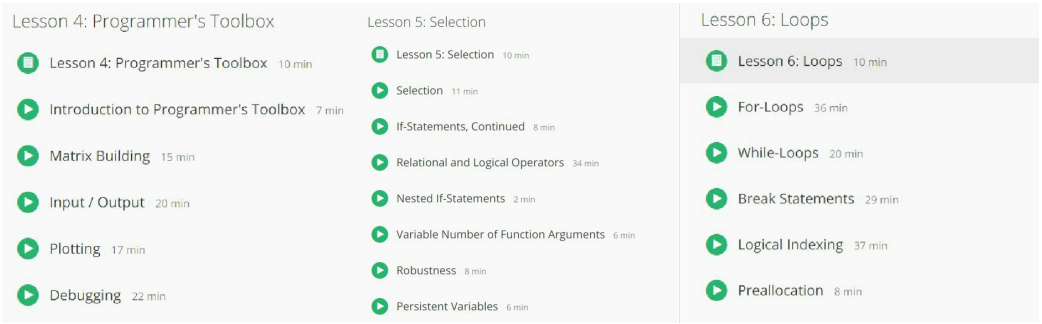

Figure 26: The Syllabus of Matlab Course for 4-5-6 Weeks

Simple sorting code I wrote as an assignment for the course can be examined at Appendix C[3].

5.2 Training on Microsoft Sharepoint

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.

In advice to our supervisor, I started to use MS Sharepoint that comes within the Office 365. Even though I have been using Office 365 student account since 2014, I did not notice Sharepoint's benefits. After seeing that all the engineers in Türksat uses Sharepoint as a document storage and share point, I started to create proper content types and site columns for my needs in school life. I created the draft version of columns and content types in Confluence wiki then transport hem to Sharepoint. These can be seen at *Figure 27*

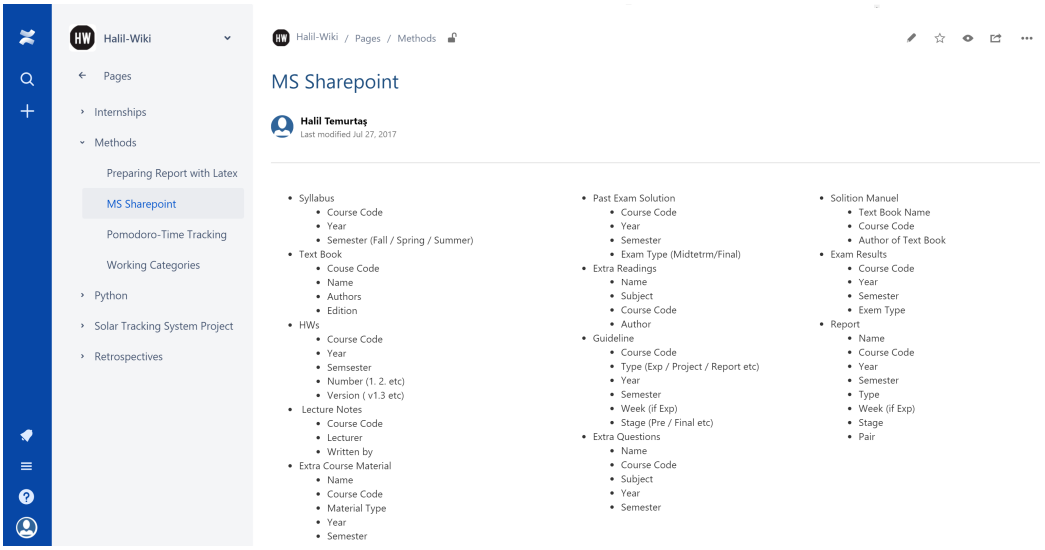

Figure 27: Columns & Content Types in Confluence Wiki

6 Conclusion

I completed my summer practice in TÜRKSAT A.Ş.(Türksat Satellite Communications and Cable TV Operations Company) in Gölbaşı/Ankara. It was quite experiential work time for me. Throughout my summer practice, I learned many things about professional work life. Firstly, I witnessed how big projects handled in a big companies like TÜRKSAT through collaboration by team members. Moreover, I took part in a small scale by building a solar tracker with team member interns from other engineering departments. Secondly, I gained experience over using project management tools. While taking advantages of these tools in our project, I witnessed and understood how valuable and essential tools for success of companies and projects. Thirdly, while doing all of this, I leant and used many useful engineering tools for proffessional life such as Python, Raspberry Pi, Matlab and so on.

Finally, I recommend my summer practice company for other students. Moreover, I strongly recommend Directorate of Satellite Programming for their summer practice if they want to observe the work done behind the project since the directorate organises the separate projects and handles their relation in the bigger projects like Türksat 6-A project. I believe, I spent my time in the internship effectivly as possible.

7 References

[1] Raspberry Pi 2 & 3 Pin Mappings, https://docs.microsoft.com/en-us/windows/iot-core/learn-about-hardware/pinmappings/pinmappingsrpi

[2] Temurtas Halil, *Sun Tracker System*, Bitbucket repository,(2017), https://bitbucket.org/temurtas/pi/

[3] Temurtas Halil, *Matlab Codes*, Bitbucket repository,(2017), https://bitbucket.org/temurtas/staj_matlab

[4] *Wiki* https://www.britannica.com/topic/wiki

[5] Temurtas Halil, *EE300 Report*, Bitbucket repository,(2017), https://bitbucket.org/temurtas/ee300_report

[6] *Pomotodo Web App*, https://pomotodo.com/app/

[7] Temurtas H.,Koculu E.,İzmir T.,Göçer A., *Sun Tracker System*, Airtable Base, (2017), https://airtable.com/shrI9Y26ehXklCe9m

[8] Temurtas H.,Koculu E.,İzmir T.,Göçer A., *HR*, Airtable Base, (2017), https://airtable.com/shrCJKhPqLuX9y0lh

[9] *Airtable Guide*, https://guide.airtable.com/

[10] Francesco Cirillo *The Pomodoro Technique* http://baomee.info/pdf/technique/1.pdf

[11] *TURSKAT About Us* https://www.turksat.com.tr/en/corporate/about-us

[12] *Coursera Matlab Course Page*, https://www.coursera.org/learn/matlab

[13] Rana L. A., *Automatic sun tracking system*, ResearchGate, (2017), https://www.researchgate.net/publication/248706918_Automatic_sun_tracking_system

[14] Syed Arsalan, *Sun Tracking System with Microcontroller 8051*, International Journal of Scientific & Engineering Research, Volume 4, Issue 6, (June-2013), https://www.ijser.org/researchpaper/Sun-Tracking-System-with-Microcontroller-8051.pdf

A 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)
```

```
44          time.sleep(1)
45      else :
46          GPIO.output(25,GPIO.LOW)
47          GPIO.output(23,GPIO.LOW)
48          time.sleep(1)
49
50      if uc>dort and fark3>0.01:
51          GPIO.output(24,GPIO.HIGH)
52          GPIO.output(8,GPIO.LOW)
53          time.sleep(1)
54      elif dort>uc and fark4>0.01:
55          GPIO.output(8,GPIO.HIGH)
56          GPIO.output(24,GPIO.LOW)
57          time.sleep(1)
58      else :
59          GPIO.output(24,GPIO.LOW)
60          GPIO.output(8,GPIO.LOW)
61          time.sleep(1)
```

B Arduino Final Code

```
#include <Servo.h>

Servo servo1;
Servo servo2;

int in_rasp1 =3;
int in_rasp2 =4;
int in_rasp3 =5;
int in_rasp4 =6;

int read1=0;
int read2=0;
int read3=0;
int read4=0;

void setup()
{
    servo1.attach(9);
    servo1.writeMicroseconds(1475);
    servo2.attach(10);
    servo2.writeMicroseconds(1475);

    pinMode(in_rasp1, INPUT);
    pinMode(in_rasp2, INPUT);
    pinMode(in_rasp3, INPUT);
    pinMode(in_rasp4, INPUT);
}
void loop()  {
    read1 =digitalRead(in_rasp1);
    read2 =digitalRead(in_rasp2);
    read3 =digitalRead(in_rasp3);
    read4 =digitalRead(in_rasp4);

    if (read1 == HIGH)
    {
        servo1.writeMicroseconds(1515);
        delay(42);
        servo1.writeMicroseconds(1475);
        delay(200);
    }
    else if (read2 == HIGH)
    {
        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  }
```

C Simple Sorting Code in MATLAB

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

if a1 <= a2
    if a2 <= a3
        a = a1
        b = a2
        c = a3
    else
        e = a3
        a3 = a2
        a2 = e

        if a1 <= a2
            a = a1
            b = a2
            c = a3
        else
            w = a2
            a2 = a1
            a1 = w
            a = a1
            b = a2
            c = a3
        end
    end
else
    w = a2
    a2 = a1
    a1 = w
    if a2 >= a3
        e = a3
        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
44                  a1 = w
45                  a = a1
46                  b = a2
47                  c = a3
48              end
49          else
50              a = a1
51              b = a2
52              c = a3
53          end
54  end
55  end
56  }
```