



# **SİSTEM PROGRAMLAMA**

**2017 - 2018 BAHAR DÖNEMİ**

**DÖNEM PROJESİ : Ev Otomasyon Sistemi**

152120141003 - Şafak AKINCI  
152120141014 - Ahmet Gökçe BOZAN

## **İÇİNDEKİLER**

### **1. PROJE AMACI**

### **2. PROJE DETAYI**

**Adım 1. Donanım Tasarımı ve Gerçeklemesi**

**Adım 2. Yazılım Tasarımı ve Gerçeklemesi**

**Adım 2. 1. Sunucu Yazılım**

**Adım 2. 2. Sunucu Arayüz Yazılımı**

**Adım 2. 3. Uygulama Yazılımı**

### **UYGULAMA ÇIKTILARI**

## 1 - PROJE AMACI:

Raspberry Pi 3'ü, ev otomasyonunda kullanılacak bir alt sistem haline getirmektir. Ev otomasyonunda ihtiyaç duyulan algılayıcı( sensörler) ve sürücüler ( röle ) doğrudan Raspberry Pi 3'e bağlanacaktır. Bu sistem bir sunucu modül gibi çalışarak, üst seviyede yazılan uygulamalara ağ üzerinden ve seri bağlantı üzerinden algılayıcı ve sürücülere erişim imkanı sağlayacaktır.

## 2 - PROJE DETAYI:

### Adım 1. Donanım Tasarımı ve Gerçekleşmesi:

Tasarımda, aşağıda belirtilen giriş çıkış birimleri kullanılmıştır.

İkilik sistemde ( 0 veya 1 ) çıktı üreten sensörler:

Sürücü olarak:

Klavye olarak:

**Eğim Sensörü, Mesafe Sensörü**

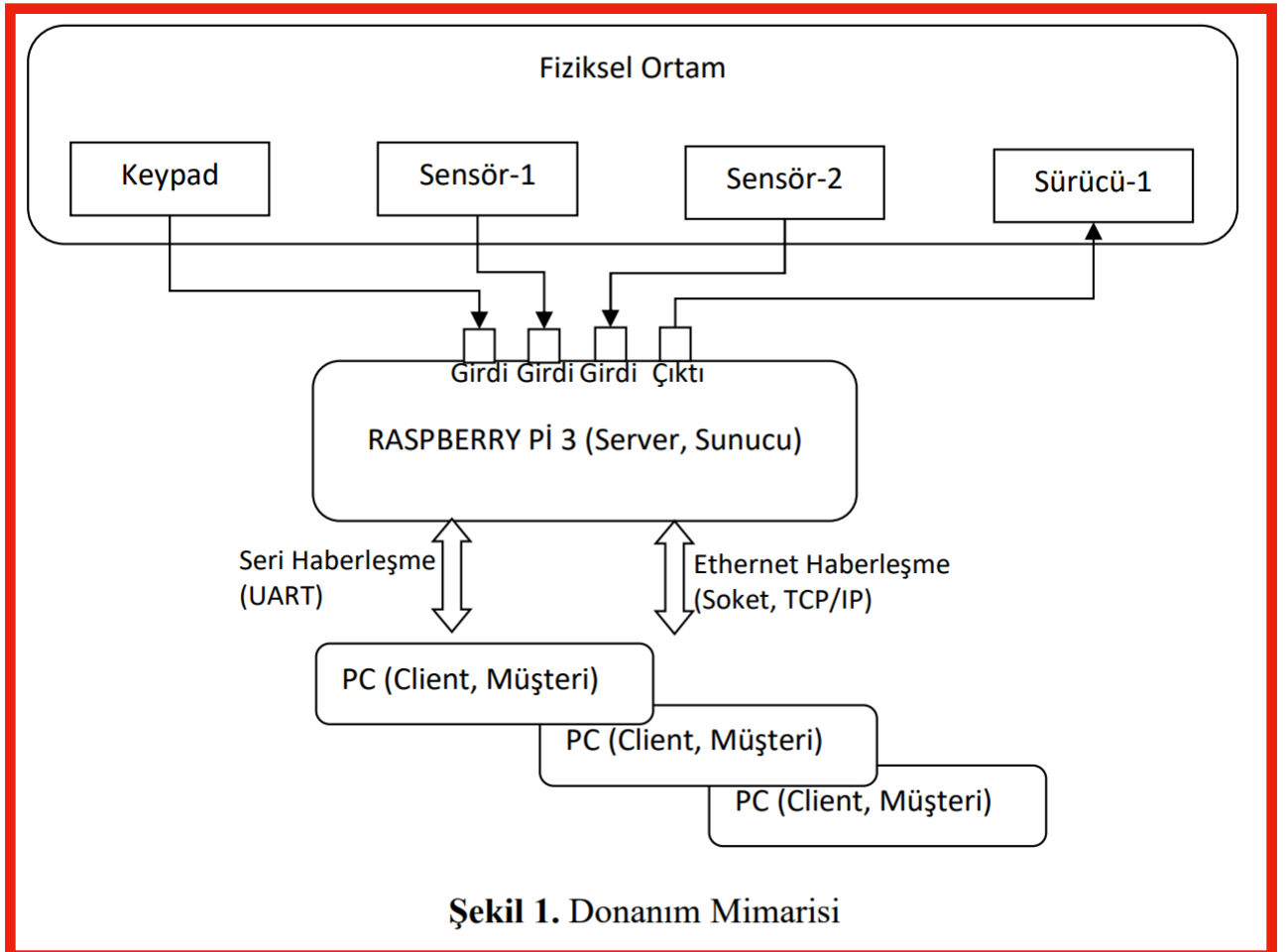
**Röle**

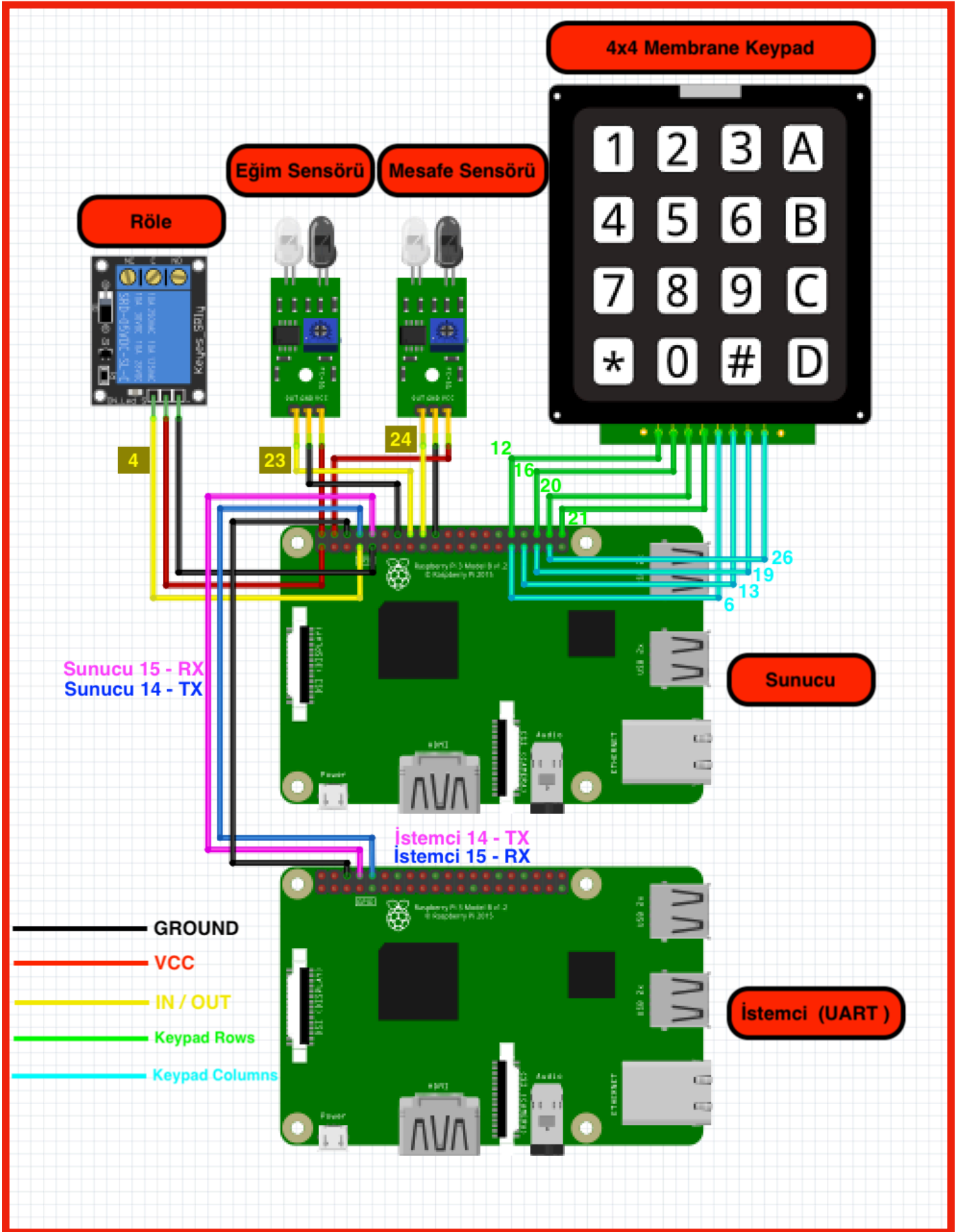
**4x4 Membrane Keypad**

**Eğim sensörü**, eğimi algılayabilmek için içlerinde civa damlacığı ya da metal bilye bulundurur. Bu sensörler bulundukları konuma göre içlerindeki civa damlacığının ya da metal bilyenin sensör içerisindeki anahtarları açması ya da kapamasıyla çalışır.

**Mesafe sensörleri**, gelişmiş yansımali sensörlerdir. Bu sensörler bir IR verici led tarafından yayılan IR ışığın IR alıcı modül ile toplanması mantığıyla çalışırlar. IR ışınların yayılımı kodlanmıştır ve kesik kesiktir.

**Röle**, düşük akımlar kullanarak yüksek akım çeken cihazları anahtarlama görevinde kullanılan devre elemanıdır. Rölenin bobinine enerji verildiğinde mıknatıslanan bobin bir armatürü hareket ettirerek kontakların birbirine temasını sağlar ve devrede iletim sağlamış olur. Örnek kullanım alanları sigortalar, alarm sistemleri, elektrikli ev aletleri vb.



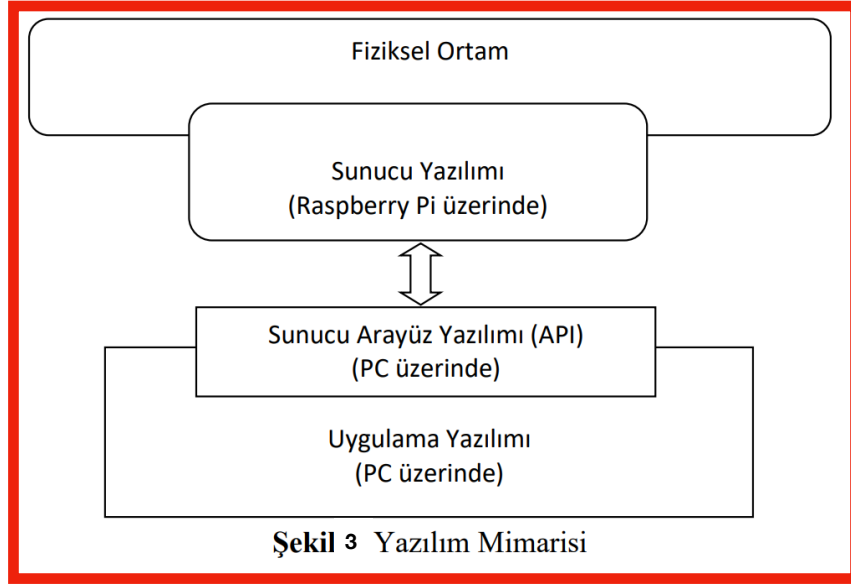


Şekil 2: Donanım Tasarımı ve Gerçeklemesi

## Adım 2. Yazılım Tasarımı ve Gerçeklemesi

Şekil 3'de gösterildiği gibi, sistemde üç farklı yazılım geliştirilmiştir.

1. Raspberry Pi üzerinde koşacak sunucu yazılımı.
2. Raspberry Pi'ye Ethernet (Socket) ya da Seri Port ( UART ) üzerinden erişecek olan, PC tarafından uygulama geliştirmede kullanılacak olan kütüphane. Raspberry pi ile haberleşme ve mesaj denetimleri bu kütüphanedeki fonksiyonlarla sağlanacaktır.
3. İlk iki yazılımı test etmek üzere çalıştırılacak uygulama yazılımıdır.



### Adım 2. 1. Sunucu Yazılım

- Sürücü için açma ya da kapama talebi geldiğinde, buna göre sürücünün durumu değiştirilecektir.

İstemcilerden gelen istekleri bağlantı türüne göre ( UART ya da SOKET) dinlemek için iki adet iş parçacığı kullanılmaktadır. ( **threadReadUART**, **threadCheckSocketMessage** )

- Program çalışması ile birlikte, soket yaratarak bağlantı istekleri için bekleyecek, bağlantı isteği alırsa bağlantıyı kuracak, bağlantı kesildiğinde yeni bir bağlantı için beklemeye devam edecektir.

Bir iş parçacığı ( **threadMakeConnection** ) sürekli olarak, soket üzerinden gelen bağlantı isteklerini dinler ve kabul eder ( Sunucuya en fazla 5 istemci bağlanabilir. )

- Sensörler periyodik olarak okunarak, lokal değişkenlerde güncel değerleri tutulacaktır.

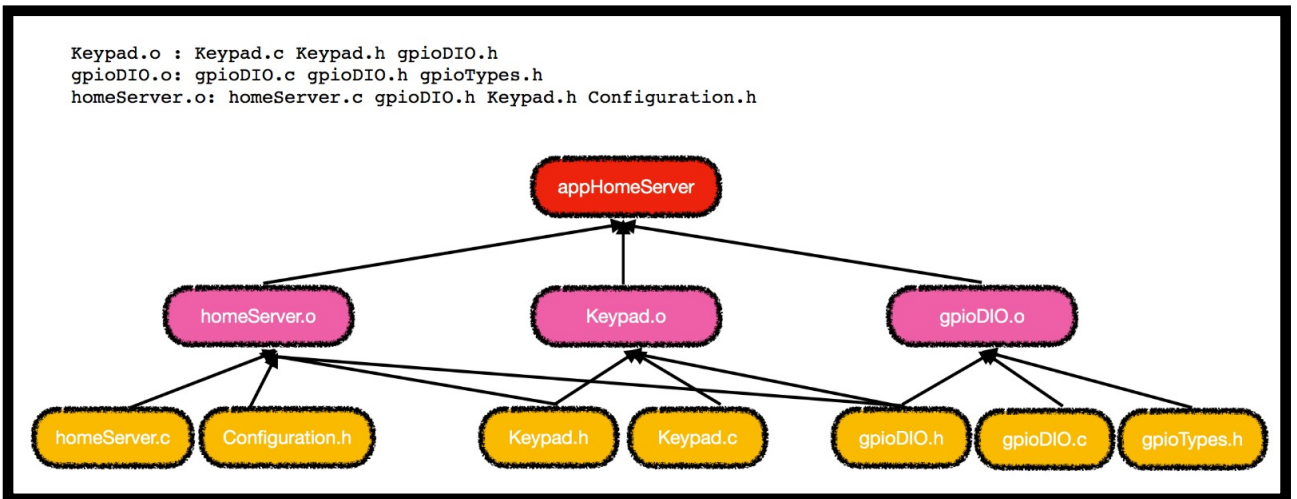
Bir iş parçacığı ( **threadReadSensorValues** ) sürekli olarak, sensörlerden gelen değerleri okur ve lokal değişkenlerde tutar.

- Keypad sürekli dinlenecektir ( Kapı şifresi gibi düşünülebilir). Keypad üzerinden, şifre değişimi ve şifre ile giriş yaparak role açma ve kapama işlemi yapılabilecektir. Buna göre; XXXX\*YYYY# ( tuşlandığında, XXXX doğru şifre ise, YYYY olarak değiştirilecektir.) XXXX\*0# ( tuşlandığında, XXXX doğru şifre ise, röle kapatılacaktır.) XXXX\*I# ( tuşlandığında, XXXX doğru şifre ise, röle açılacaktır.) Sistem ilk çalıştığında şifre 0000 olarak tanılacaktır. Kullanıcı isterse, şifreyi istediği bir şifre ile yukarıdaki tuşlamaları kullanarak gerçekleştirecektir.

Bir iş parçacığı, ( **threadListenKeypad** ) sürekli olarak, sunucuya bağlı olan keypadi dinler ve gerekli işlemleri yapar.

ALINAN MESAJ	CEVAP MESAJI	EYLEM
sensorDurum:	sensorDurum <durum>:	Güncel sensör durumları paketlenerek gönderilir.
sensorSayi:	sensorSayi <sensor sayisi>:	Sunucuya bağlı toplam sensör sayısını gönderir.
sensorList:	sensorList <ad1, ad2>:	Sunucuya bağlı sensör isimlerini gönderir.
surucu <0 ya da 1>:	surucu <ok veya error>:	Sürücü istenen duruma getirilir ve istemciye durum bilgisi iletilir.
surucuDurum:	surucuDurum <0 veya 1>:	Sürücünün güncel durumu istemciye iletilir.
autoMod <on sensorId sensorDeğeri>:	autoMod <ON>:	İstemci, otomatik moddan çıkana kadar, belirtilen sensörün durumuna göre röle kontrolü yapılır.
autoMod <off>:	autoMod <OFF>:	Otomatik moddan çıkıldığı istemciye iletir.
close:	close <OK>:	Sunucu ile istemci arasındaki bağlantı kapatılır.

Sunucu yazılımının derlenmesi için gerekli olan makefile dosyasını bağıllık ağacı Şekil 4'de gösterilmiştir.



Şekil 4: Server Makefile

## Adım 2. 2. Sunucu Arayüz Yazılımı

Sunucu ile istemci arasında iletişimi sağlayacak kütüphane yazılmıştır ve raporla beraber sunulacaktır. Aşağıda belirtilen dosyaların içinde, iletişimi sağlayacak gerekli fonksiyonlar kodlanmıştır.

CommunicationAPI.c  
CommunicationAPI.h

Sunucu arayüz yazılımının terminal üzerinden "**libCommunicationAPI.so**" isimli dinamik kütüphane olarak derlenmesi için gerekli olan adımlar aşağıdaki gibidir.

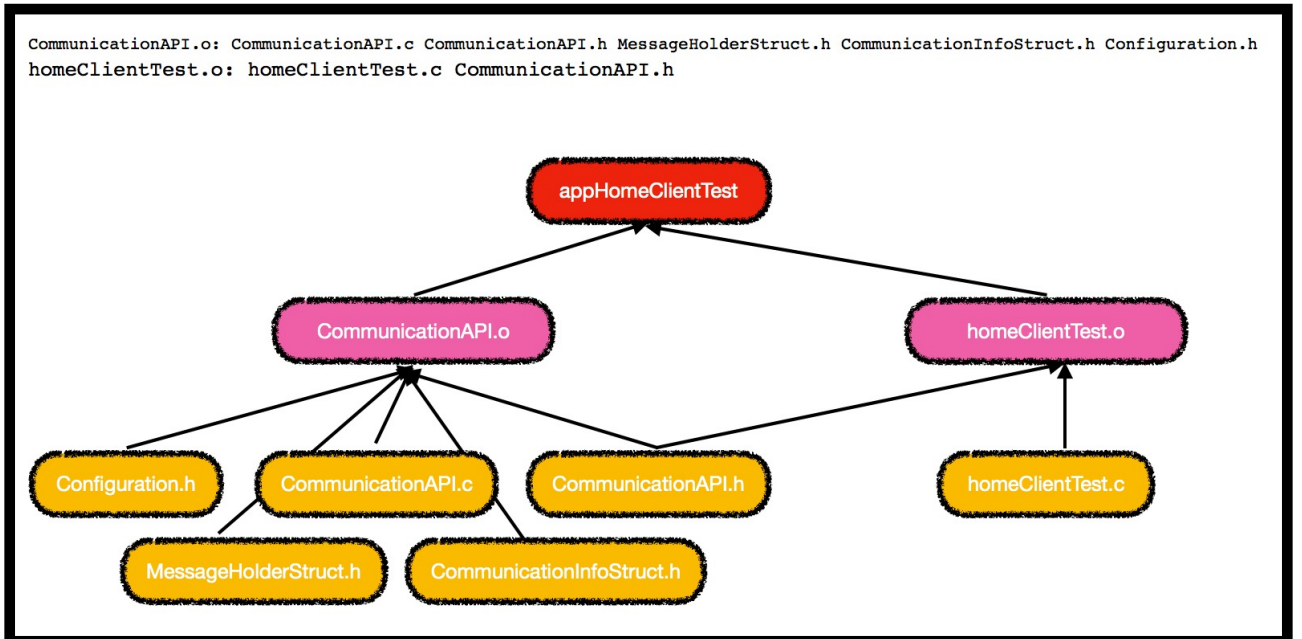
```
$ gcc -c -Wall -fpic CommunicationAPI.c -o CommunicationAPI.o
$ gcc -shared -o libCommunicationAPI.so CommunicationAPI.o
$ export LD_LIBRARY_PATH=/home/pi:$LD_LIBRARY_PATH
$ gcc -L./ -l./ -Wall -o appHomeClientTest homeClientTest.c -lCommunicationAPI
```

## Adım 2. 3. Uygulama Yazılımı

Sunucu ile geliştirilen arayüz yazılımı kullanılarak oluşturulan "appHomeClientTest" uygulama yazılımı, kullanıcıya bir menü sunarak aşağıda listelenen işlemleri gerçekleştirecektir.

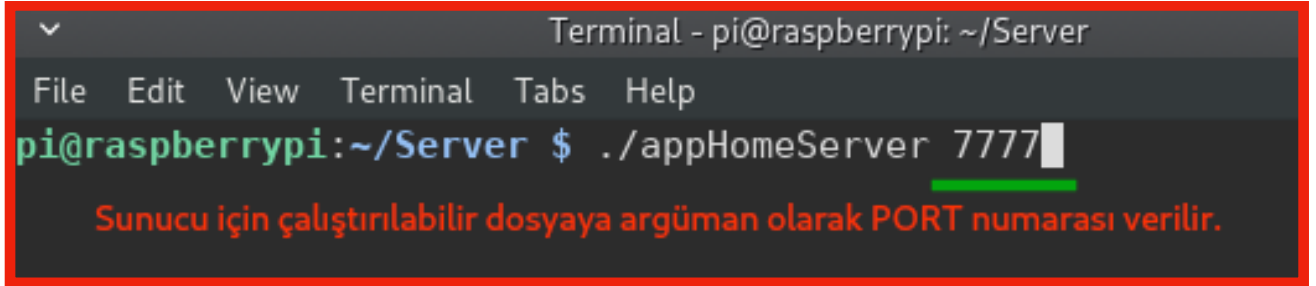
1. Güncel sensör değerlerinin öğrenilmesi
2. Toplam sensör sayısının öğrenilmesi
3. Sunucuda mevcut olan tüm sensörlerin isimleri ve id'lerinin listelenmesi
4. Röle durumunun değiştirilmesi
5. Güncel röle durumunun öğrenilmesi
6. Automod'a geçilmesi
7. Sunucu ile bağlantının kapatılması ve programın sonlandırılması

Uygulama yazılımı için gerekli olan makefile dosyasına ait bağımlılık ağacı Şekil 5'de gösterilmiştir.



Şekil 5 : Client Makefile

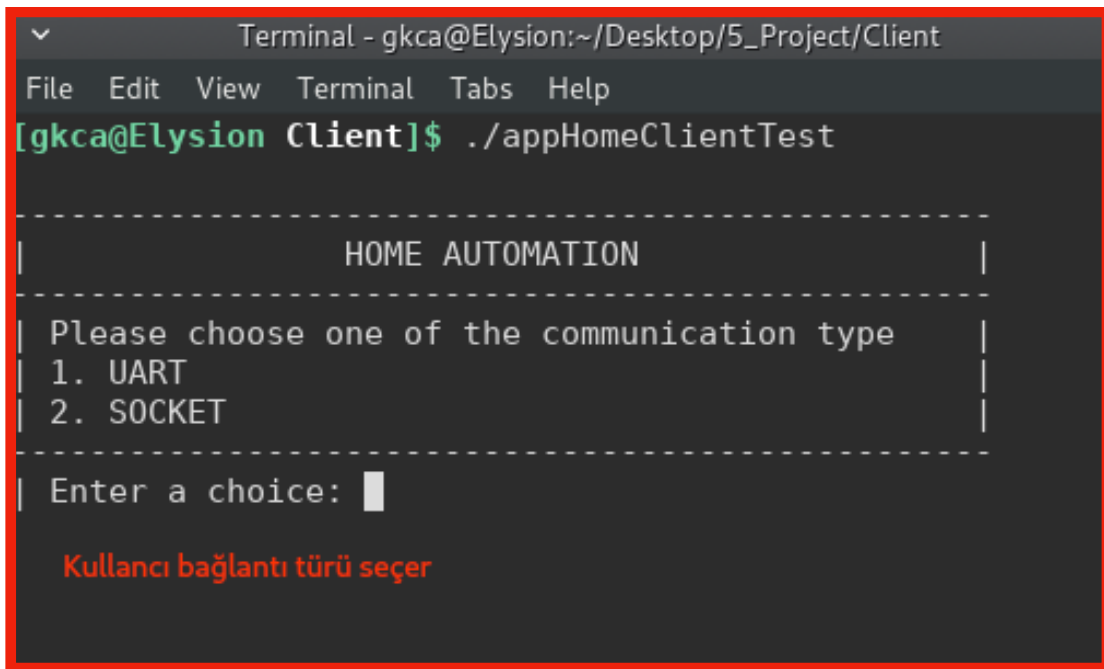
## UYGULAMA ÇIKTILARI



```
Terminal - pi@raspberrypi: ~/Server
File Edit View Terminal Tabs Help
pi@raspberrypi:~/Server $ ./appHomeServer 7777
```

Sunucu için çalıştırılabilir dosyaya argüman olarak PORT numarası verilir.

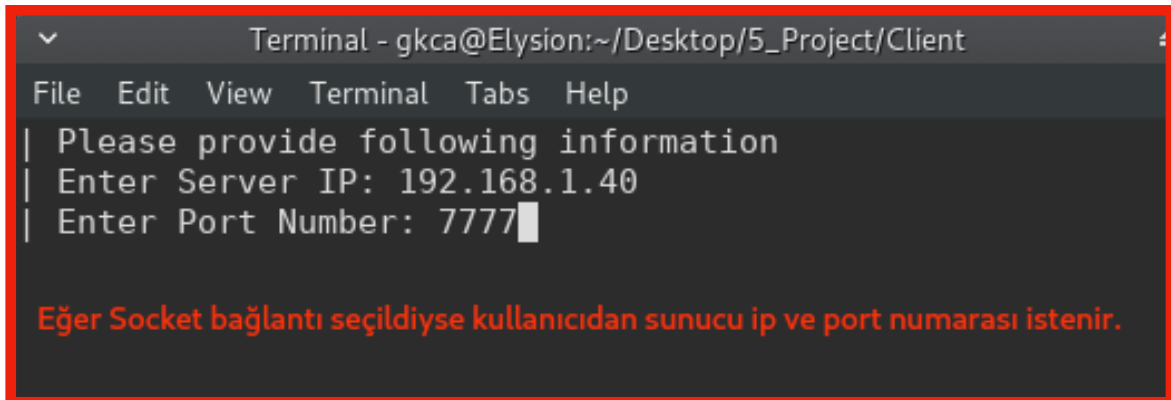
Şekil 6 : Sunucu Başlatma



```
Terminal - gkca@Elytion:~/Desktop/5_Project/Client
File Edit View Terminal Tabs Help
[gkca@Elytion Client]$ ./appHomeClientTest
-----
|                               HOME AUTOMATION                               |
|-----|
| Please choose one of the communication type |
| 1. UART |
| 2. SOCKET |
|-----|
| Enter a choice: |
```

Kullanıcı bağlantı türü seçer

Şekil 7: Kullanıcı Başlatma

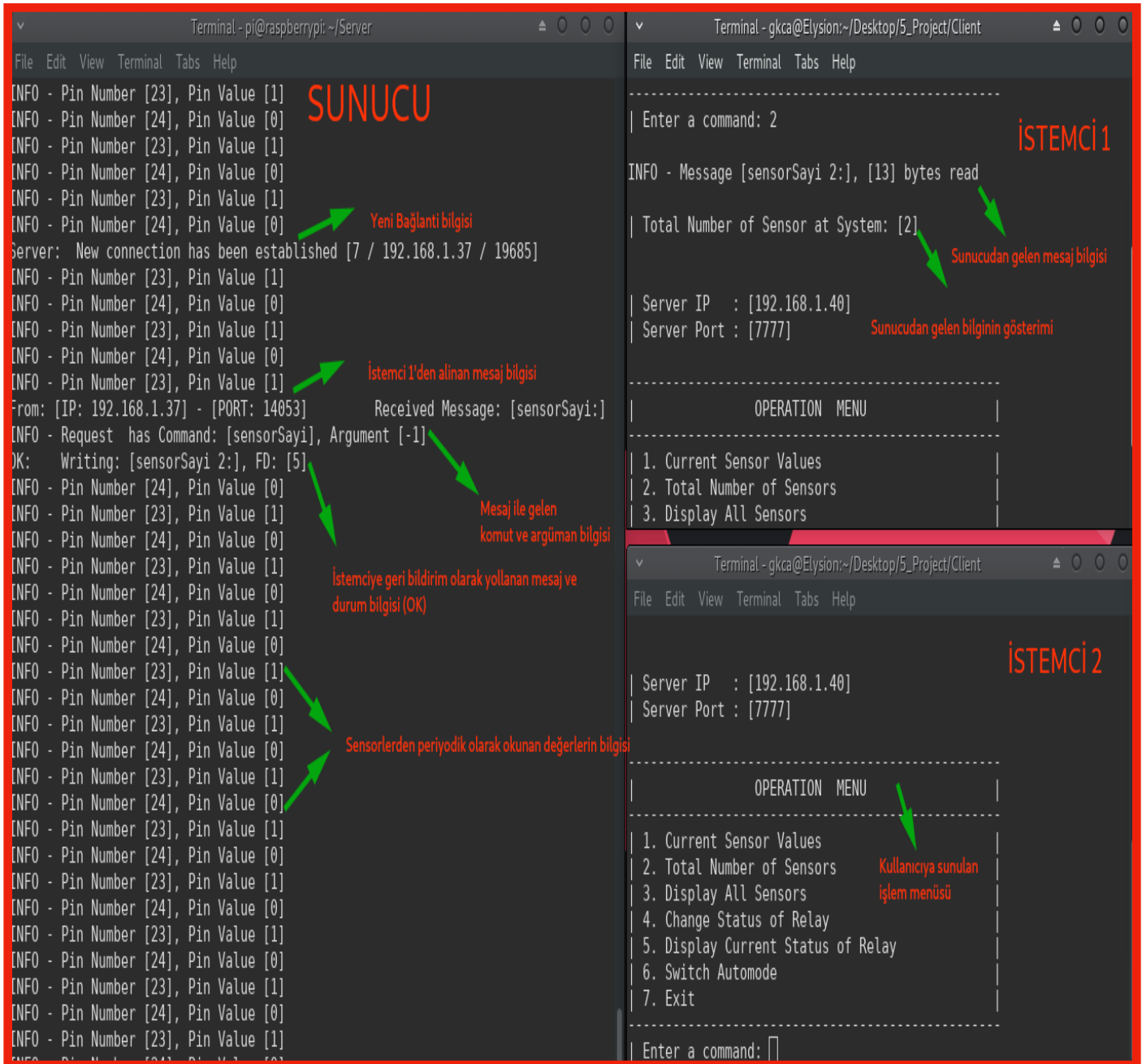


```
Terminal - gkca@Elytion:~/Desktop/5_Project/Client
File Edit View Terminal Tabs Help
| Please provide following information
| Enter Server IP: 192.168.1.40
| Enter Port Number: 7777
```

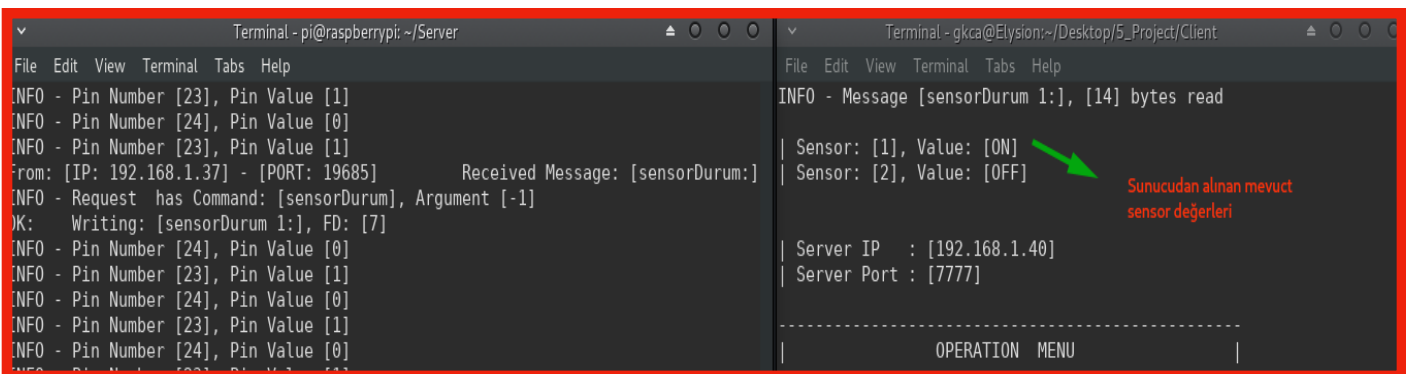
Eğer Socket bağlantı seçildiyse kullanıcıdan sunucu ip ve port numarası istenir.

Şekil 8: Socket Başlatma





### Şekil 9 : Sunucu ve İki İstemci



### Şekil 10: Güncel Sensör Değerlerinin Öğrenilmesi

```

Terminal - pi@raspberrypi: ~/Server
File Edit View Terminal Tabs Help
INFO - Pin Number [23], Pin Value [1]
INFO - Pin Number [24], Pin Value [0]
INFO - Pin Number [23], Pin Value [1]
INFO - Pin Number [24], Pin Value [0]
From: [IP: 192.168.1.37] - [PORT: 19685] Received Message: [sensorSayi:]
INFO - Request has Command: [sensorSayi], Argument [-1]
OK: Writing: [sensorSayi 2:], FD: [7]
INFO - Pin Number [23], Pin Value [1]
INFO - Pin Number [24], Pin Value [0]
INFO - Pin Number [23], Pin Value [1]

Terminal - gkca@Elysiom: ~/Desktop/5_Project/Client
File Edit View Terminal Tabs Help
INFO - Message [sensorSayi 2:], [13] bytes read
| Total Number of Sensor at System: [2]
| Server IP : [192.168.1.40]
| Server Port : [7777]
-----
OPERATION MENU

```

Şekil 11: Toplam sensör sayısının öğrenilmesi

```

Terminal - pi@raspberrypi: ~/Server
File Edit View Terminal Tabs Help
INFO - Pin Number [24], Pin Value [0]
INFO - Pin Number [23], Pin Value [1]
INFO - Pin Number [24], Pin Value [0]
From: [IP: 192.168.1.37] - [PORT: 19685] Received Message: [sensorList:]
INFO - Request has Command: [sensorList], Argument [-1]
INFO - Prepared SensorList: [Egim,Mesafe:]
OK: Writing: [Egim,Mesafe:], FD: [7]
INFO - Pin Number [23], Pin Value [1]
INFO - Pin Number [24], Pin Value [0]

Terminal - gkca@Elysiom: ~/Desktop/5_Project/Client
File Edit View Terminal Tabs Help
| SENSOR LIST
| [0] : Egim
| [1] : Mesafe
| Server IP : [192.168.1.40]
| Server Port : [7777]
-----
OPERATION MENU

```

Şekil 12: Sunucuda mevcut olan tüm sensörlerin isimleri ve id'lerinin listelenmesi

```

File Edit View Terminal Tabs Help
| Set Relay by typing '1'--> ON or '0' --> OFF
| Enter a choice:1

```

Şekil 13: Röle durumunun değiştirilmesi

```

Terminal - pi@raspberrypi: ~/Server
File Edit View Terminal Tabs Help
INFO - Pin Number [23], Pin Value [1]
INFO - Pin Number [24], Pin Value [0]
From: [IP: 192.168.1.37] - [PORT: 19685] Received Message: [surucu 0:]
INFO - Request has Command: [surucu], Argument [0]
OK: Writing: [surucu OK:], FD: [7]
INFO - Pin Number [23], Pin Value [1]
INFO - Pin Number [24], Pin Value [0]
INFO - Pin Number [23], Pin Value [1]
INFO - Pin Number [24], Pin Value [0]
INFO - Pin Number [23], Pin Value [1]
INFO - Pin Number [24], Pin Value [0]

Terminal - gkca@Elysiom: ~/Desktop/5_Project/Client
File Edit View Terminal Tabs Help
| Enter a choice:1
INFO - Message [surucu OK:], [10] bytes read
| [OK] - Relay Status Change
| Server IP : [192.168.1.40]
| Server Port : [7777]
-----
OPERATION MENU

```

Şekil 14: Röle durumunun değiştirilmesi

```

Terminal - pi@raspberrypi: ~/Server
File Edit View Terminal Tabs Help
INFO - Pin Number [24], Pin Value [0]
INFO - Pin Number [23], Pin Value [1]
INFO - Pin Number [24], Pin Value [0]
From: [IP: 192.168.1.37] - [PORT: 19685] Received Message: [surucuDurum:]
INFO - Request has Command: [surucuDurum], Argument [-1]
K: Writing: [surucuDurum 0:], FD: [7]
INFO - Pin Number [23], Pin Value [1]
INFO - Pin Number [24], Pin Value [0]
INFO - Pin Number [23], Pin Value [1]
INFO - Pin Number [24], Pin Value [0]

Terminal - gkca@Elysion: ~/Desktop/5_Project/Client
File Edit View Terminal Tabs Help
INFO - Message [surucuDurum 0:], [14] bytes read
| Relay: [0N]
| Server IP : [192.168.1.40]
| Server Port : [7777]
-----
OPERATTON MENU

```

Şekil 15: Güncel röle durumunun öğrenilmesi

```

Terminal - pi@raspberrypi: ~/Server
File Edit View Terminal Tabs Help
INFO - Pin Number [23], Pin Value [1]
INFO - Pin Number [24], Pin Value [0]
From: [IP: 192.168.1.37] - [PORT: 19685] Received Message: [autoMode on 1
1.]
INFO - Request has Command: [autoMode on 1 1], Argument [-1]
INFO - Automode ON!
INFO - Pin Number [23], Pin Value [1]
INFO - Pin Number [24], Pin Value [0]
INFO - Pin Number [23], Pin Value [1]
INFO - Pin Number [24], Pin Value [0]
INFO - Pin Number [23], Pin Value [1]

Terminal - gkca@Elysion: ~/Desktop/5_Project/Client
File Edit View Terminal Tabs Help
SENSOR LIST
| [0] : Egim
| [1] : Mesafe
Enter ID of Sensor:0
What Sensor Value opens the Relay? ( 1 or 0 ):1
To Exit in 'AutoMod' press 'e':

```

Şekil 16: Automod'a geçilmesi