



LOW LEVEL DESIGN AND IMPLEMENTATION DOCUMENT

IMPLEMENTATION OF PRECISION AGRICULTURE MONITORING SYSTEM USING RASPBERRY PI AND CROP PREDICTION USING MACHINE LEARNING ALGORITHM

UE17CS490B – Capstone Project Phase – 2

Submitted by:

Amala	01FB17ECS703
Anusha B	PES1201701061
Sharada G	PES1201802412
Veena K	PES1201802492

Under the guidance of

Prof. CHARANRAJ.B. R
Assistant professor
PES University

January - May 2021

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
FACULTY OF ENGINEERING
PES UNIVERSITY**

(Established under Karnataka Act No. 16 of 2013)

100ft Ring Road, Bengaluru – 560 085, Karnataka, India

TABLE OF CONTENTS

1. Introduction	4
1.1 Overview	4
1.2 Purpose	4
2. Design Considerations, Assumptions and Dependencies	4
3. Design Description	4
3.1 Module 1	4
3.1.1 Description	4
3.1.2 Use Case Diagram	4
3.1.3 Class Diagram	5
3.1.4 Sequence Diagram	7
4. Proposed Methodology / Approach	9
4.1 Algorithm and Pseudocode	9
4.2 Implementation and Results	9
Appendix: References	9

Note:

Section 1	Common for Prototype/Product Based and Research Projects
Section 2 & 3	Applicable for Prototype / Product Based Projects.
Section 4	Applicable for Research Projects.
Appendix	Provide details appropriately

1. Introduction

1.1. Overview

This project is about precision agriculture monitoring system using raspberry pi. We are using sensors which senses the parameters details, those details are sent to think speak for visualization purpose. Sensors used are temperature and humidity, fire sensor, soil moisture, Relay for pumping water to fields whenever soil moisture content losses it moisture or value less than a fixed value of moisture. similarly, for temperature and humidity when temperature increases it temperature and humidity loses it humidity content then user gets the message and automatically pump gets on.

Here we are describing about the low level design in which we have divide the project in sublevel so that every processing steps are explained clearly such as class diagram, use case diagram sequence diagram, package development etc. These give us the overview of the project what we are going to do.

1.2. Purpose

Purpose of use precision agriculture is to get the accurate value of the parameters such as soil moisture sensor, humidity & temperature sensors and fire sensor for detection of fire in the farm. In low level design we see the progress of small part of the project, sequence diagram, use case diagram.

2. Design Constraints, Assumptions, and Dependencies

Constraints:

Network feasibility for GSM and things speak and productivity may or may not be more.
We cannot estimate weather conditions as pollution is increasing gradually etc.

Software dependencies:

- Pycharm IDE
- Thing speak
- Fast2SMS

Hardware dependencies:

- Raspberry pi3
- DTH sensor
- Soil moisture sensor
- Relay
- Pump
- Power Supply

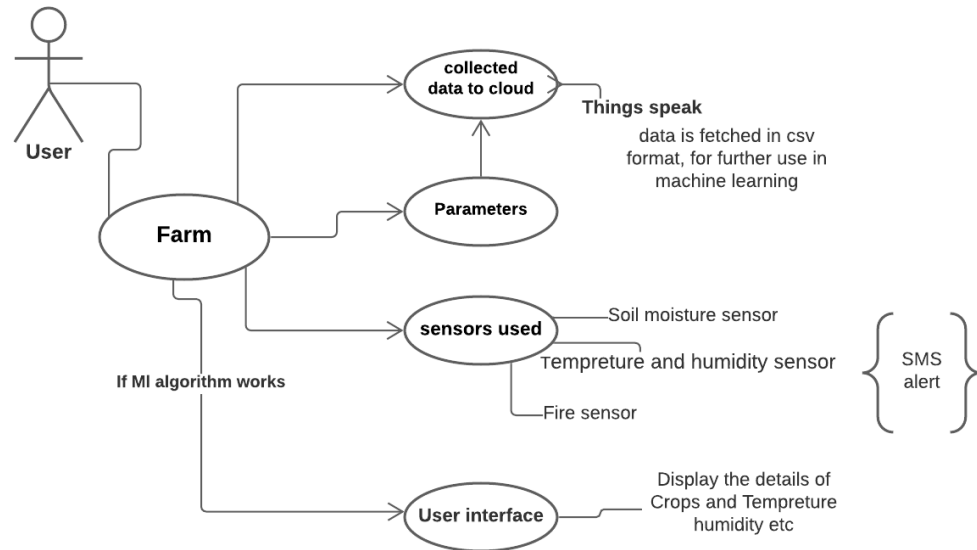
3 Master Class Diagram

3.1.1 Description

In this, whenever the user gets details from the farm those data are stored in the cloud using IOT i.e., Thing speak for now we are using a dataset from Kaggle website based on these data we are trying to predict what type of crop can be grown in these conditions.

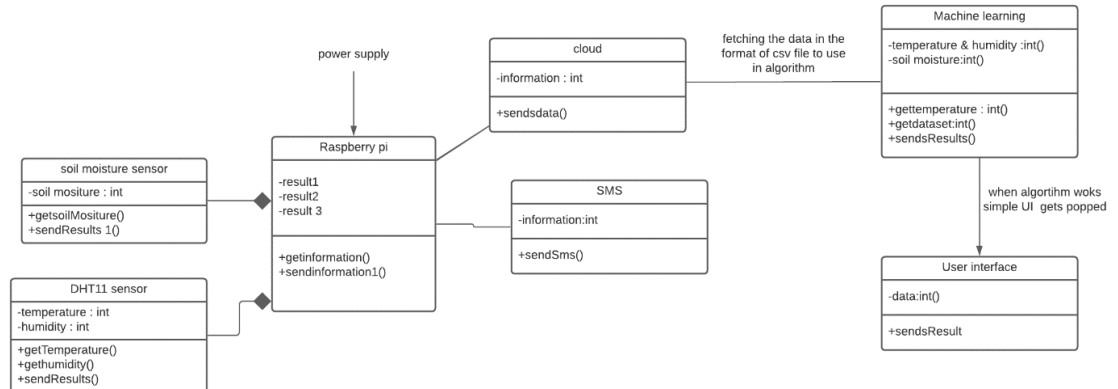
In real time we can take readings every day and can be stored in Thing speak and then after few months, we can extract the files in CSV/XML format which was taken every day based on that we can also predict which type of crop is growing there to that suitable condition. When the algorithm takes data and display in simple UI. This dependent on the algorithm.

3.1.2 Use Case Diagram

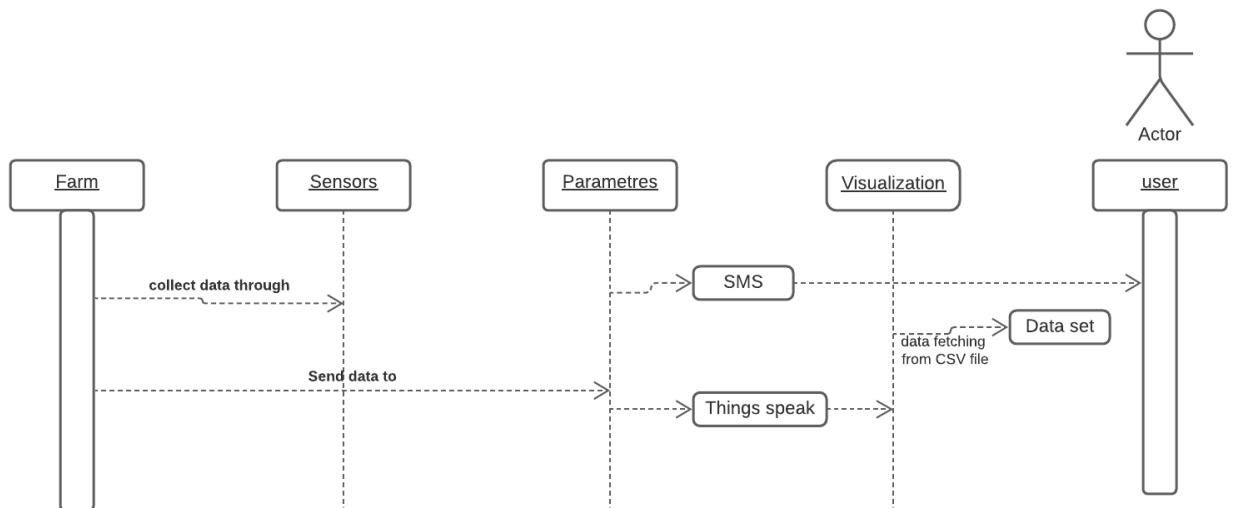


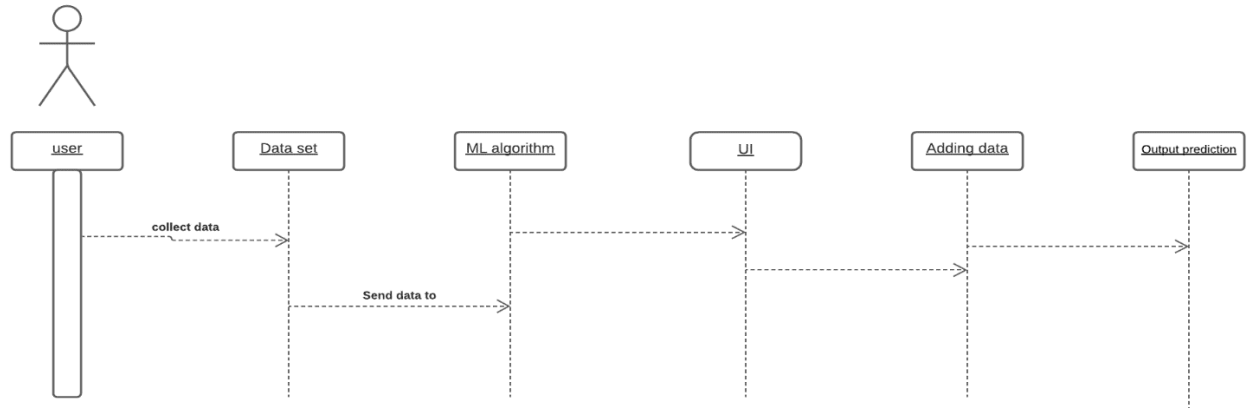
Use case Item	Description
User	User can control the sensor using mobile when get alert sms whenever there is fluctuation in the field
Thing speak	It is used to store data based on daily reading and can get visualization of data as per the requirement.
Algorithm	Process where the data is taken and processed such as classification, splitting the data.
GUI	User interface, in this it is just a simple UI when the algorithm works properly it executes and parameters details should be give so that it will predicts which crop can be grown in that suitable conditions

3.1.3 Class Diagram



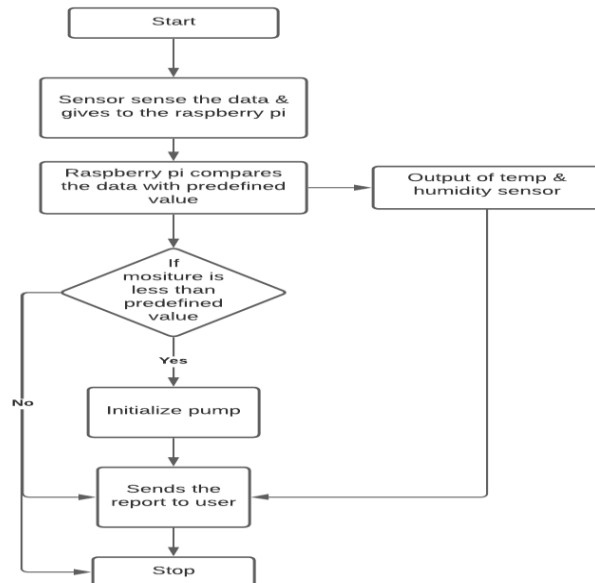
3.1.4 Sequence Diagram





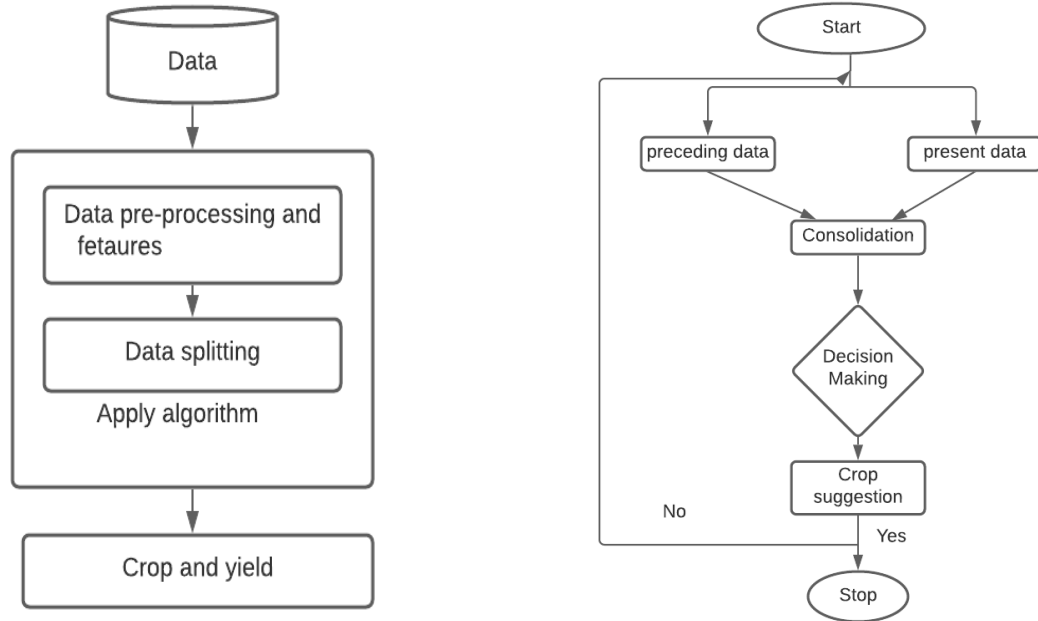
- Hardware should be connected properly to the respective pin connection.
- Sensor will sense the data and those are collected into the cloud that is Things peak is an IOT platform then visualization is done using those data and then dataset is fetched in the format of CSV/XML then further use in algorithm.
- If in case there is any change in the parameter detected value and predefined value, then SMS will be sent to the user saying that increase/decrease in temperature, humidity similarly soil moisture loses its moisture content then user gets the SMS saying that there is no water detected and automatically pump will get ON.
- Raspberry Pi OS installation process is properly done and dump the code to SD card then the System will start.
- After all the connections done sensors will sense.
- The dataset which is fetched in Things peak can be used now in the algorithm (for now we had taken the dataset from Kaggle website).
- If algorithm works properly simple UI will pop up, which is done using Tkinter and the data should be added to that required field.
- Then it predicts the Output

4 Proposed Methodology / Approach

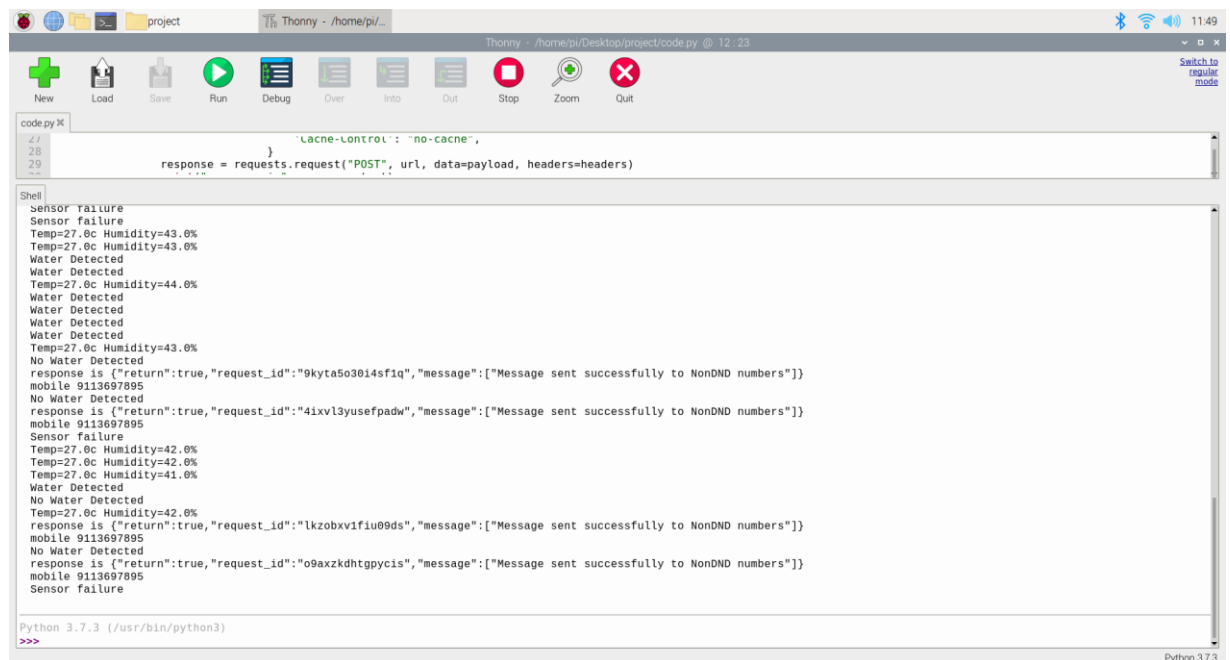


To enhance the productivity of the crop there by supporting both farmer and nation we have to use the technology which estimates the quality of crop and giving suggestions. • Wireless sensor network are sensors of different types are used to collect the information of crop conditions and environmental changes these information is transmitted through network to the farmer or devices that initiates corrective action. It also helps in collecting information about conditions like weather, moisture, temperature and fertility of soil, level of water, pest detection, animal intrusion in to the field, crop growth, agriculture. The proposed model aims at developing a smart system that would provide an ideal environment for the crops. The sensors sense the soil moisture and the humidity levels. This reduces human effort to a great extent and also ensures that an optimal environment is provided for the crops thus improving crop quality

4.1 Algorithm and Pseudocode



4.2 Implementation and Results



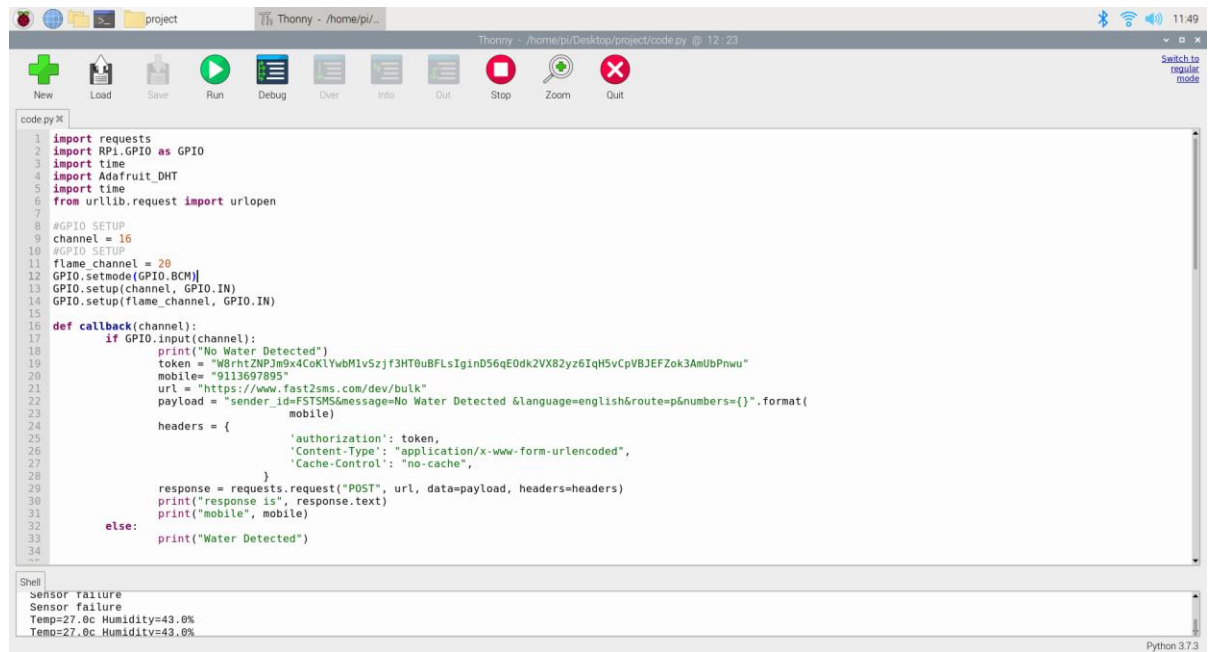
The screenshot shows the Thonny IDE interface. The code editor displays a Python script using the `requests` library to send POST requests. The terminal window shows the output of the script, including sensor failure messages, temperature and humidity readings, and successful responses from the server.

```

code.py X
28         'Lacne-Control': 'no-cacne',
29     }
    response = requests.request("POST", url, data=payload, headers=headers)

Shell
Sensor failure
Sensor failure
Temp=27.0c Humidity=43.0%
Temp=27.0c Humidity=43.0%
Water Detected
Water Detected
Temp=27.0c Humidity=44.0%
Water Detected
Water Detected
Water Detected
Water Detected
Temp=27.0c Humidity=43.0%
No Water Detected
response is {"return":true,"request_id":"9kya5o30i4sf1q","message":["Message sent successfully to NonDND numbers"]}
mobile 9113697895
No Water Detected
response is {"return":true,"request_id":"4ixv13yusefpadw","message":["Message sent successfully to NonDND numbers"]}
mobile 9113697895
Sensor failure
Temp=27.0c Humidity=42.0%
Temp=27.0c Humidity=42.0%
Temp=27.0c Humidity=41.0%
Water Detected
No Water Detected
Temp=27.0c Humidity=42.0%
response is {"return":true,"request_id":"lkzobxv1fiu09ds","message":["Message sent successfully to NonDND numbers"]}
mobile 9113697895
No Water Detected
response is {"return":true,"request_id":"o9axzkdhtgpycis","message":["Message sent successfully to NonDND numbers"]}
mobile 9113697895
Sensor failure

Python 3.7.3 (/usr/bin/python3)
>>>
  
```



```

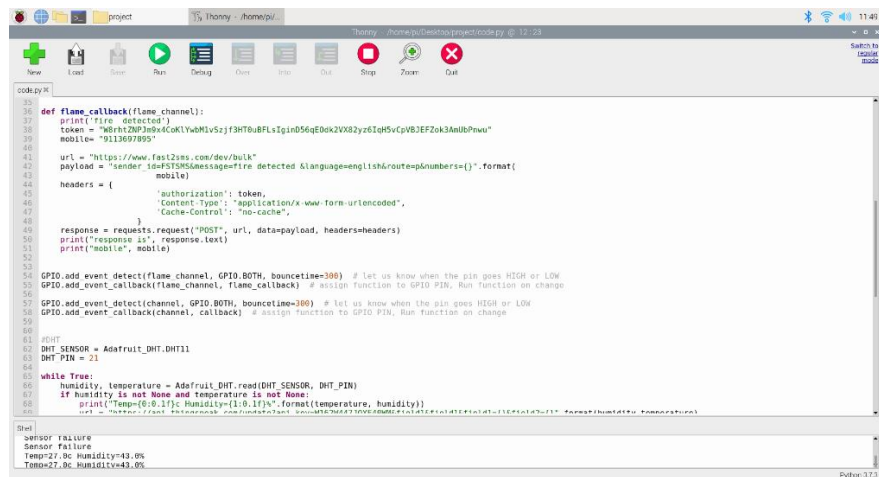
1 import requests
2 import RPi.GPIO as GPIO
3 import time
4 import Adafruit_DHT
5 import time
6 from urllib.request import urlopen
7
8 #GPIO SETUP
9 channel = 16
10 #GPIO SETUP
11 flame_channel = 20
12 GPIO.setmode(GPIO.BCM)
13 GPIO.setup(channel, GPIO.IN)
14 GPIO.setup(flame_channel, GPIO.IN)
15
16 def callback(channel):
17     if GPIO.input(channel):
18         print("No Water Detected")
19         token = "W8rhtZNP3n9x4CokLYwbMiv5zjf3HT0uBFLsIginD56qE0dk2VX82yz6IqH5vCpVB3EFZok3AmUbPnwu"
20         mobile = "9113697895"
21         url = "https://www.fast2sms.com/dev/bulk"
22         payload = "sender_id=FSTSMS&message=No Water Detected &language=english&route=p&numbers={}".format(
23             mobile)
24         headers = {
25             'authorization': token,
26             'Content-Type': "application/x-www-form-urlencoded",
27             'Cache-Control': "no-cache",
28         }
29         response = requests.request("POST", url, data=payload, headers=headers)
30         print("response is", response.text)
31         print("mobile", mobile)
32     else:
33         print("Water Detected")
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100

```

```

Shell
Sensor failure
Sensor failure
Temp=27.0c Humidity=43.0%
Temp=27.0c Humidity=43.0%

```



```

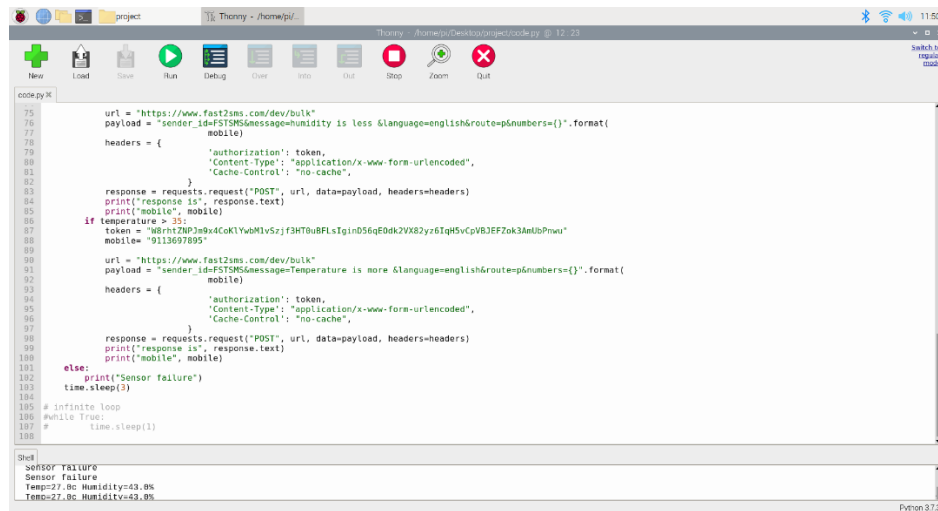
36 def flame_callback(flame_channel):
37     print("fire detected")
38     token = "W8rhtZNP3n9x4CokLYwbMiv5zjf3HT0uBFLsIginD56qE0dk2VX82yz6IqH5vCpVB3EFZok3AmUbPnwu"
39     mobile = "9113697895"
40     url = "https://www.fast2sms.com/dev/bulk"
41     payload = "sender_id=FSTSMS&message=fire detected &language=english&route=p&numbers={}".format(
42         mobile)
43     headers = {
44         'authorization': token,
45         'Content-Type': "application/x-www-form-urlencoded",
46         'Cache-Control': "no-cache",
47     }
48     response = requests.request("POST", url, data=payload, headers=headers)
49     print("response is", response.text)
50     print("mobile", mobile)
51
52
53 GPIO.add_event_detect(flame_channel, GPIO.BOTH, bounce_time=300) # let us know when the pin goes HIGH or LOW
54 GPIO.add_event_callback(flame_channel, flame_callback) # assign function to GPIO PIN, Run function on change
55
56 GPIO.add_event_detect(channel, GPIO.BOTH, bounce_time=300) # let us know when the pin goes HIGH or LOW
57 GPIO.add_event_callback(channel, callback) # assign function to GPIO PIN, Run function on change
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100

```

```

Shell
Sensor failure
Sensor failure
Temp=27.0c Humidity=43.0%
Temp=27.0c Humidity=43.0%

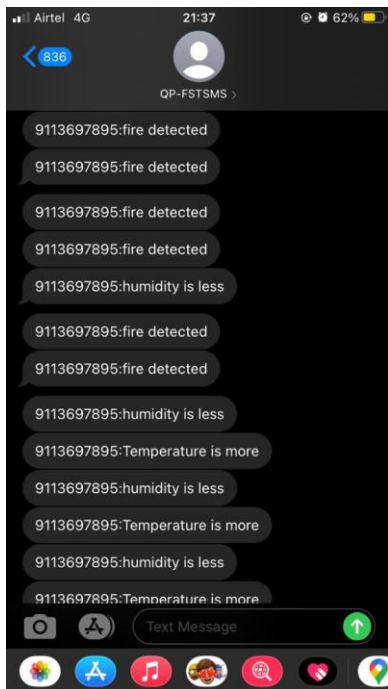
```



```

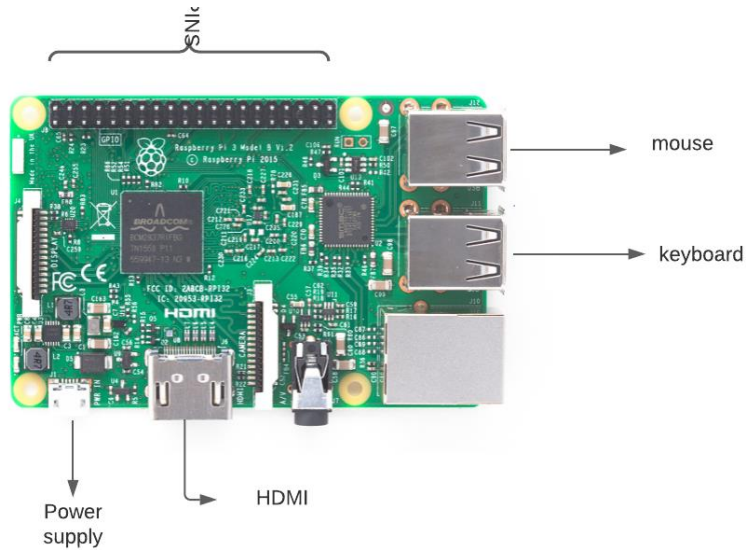
75 url = "https://www.fast2sms.com/dev/bulk"
76 payload = "sender_id=FSTSMS&message=humidity is less &language=english&route=p&numbers={}".format(
77     mobile)
78 headers = {
79     'authorization': token,
80     'Content-Type': 'application/x-www-form-urlencoded',
81     'Cache-Control': 'no-cache',
82 }
83 response = requests.request("POST", url, data=payload, headers=headers)
84 print("response is", response.text)
85 print("mobile", mobile)
86 if temperature > 35:
87     token = "W8rntZNPJe9x4CokLYvbMlvSzjF3HT0u8FLsIginD56qE0dk2VX82yz6IqH5VcpVBjEF2ok3AmUbPmwu"
88     mobile = "9113697895"
89     url = "https://www.fast2sms.com/dev/bulk"
90     payload = "sender_id=FSTSMS&message=Temperature is more &language=english&route=p&numbers={}".format(
91         mobile)
92     headers = {
93         'authorization': token,
94         'Content-Type': 'application/x-www-form-urlencoded',
95         'Cache-Control': 'no-cache',
96     }
97     response = requests.request("POST", url, data=payload, headers=headers)
98     print("response is", response.text)
99     print("mobile", mobile)
100 else:
101     print("Sensor failure")
102     time.sleep(3)
103
104 # infinite loop
105 while True:
106     # time.sleep(1)
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000

```



LOW LEVEL DESIGN AND IMPLEMENTATION DOCUMENT

HARDWARE CONNECTION:



PIN CONNECTION

1)TEMPERATURE AND HUMIDITY

-VCC->5V

-SIG->GPIO21

-GND->GND

2)FLAME SENSOR

-VCC->5V

-SIG->GPIO20

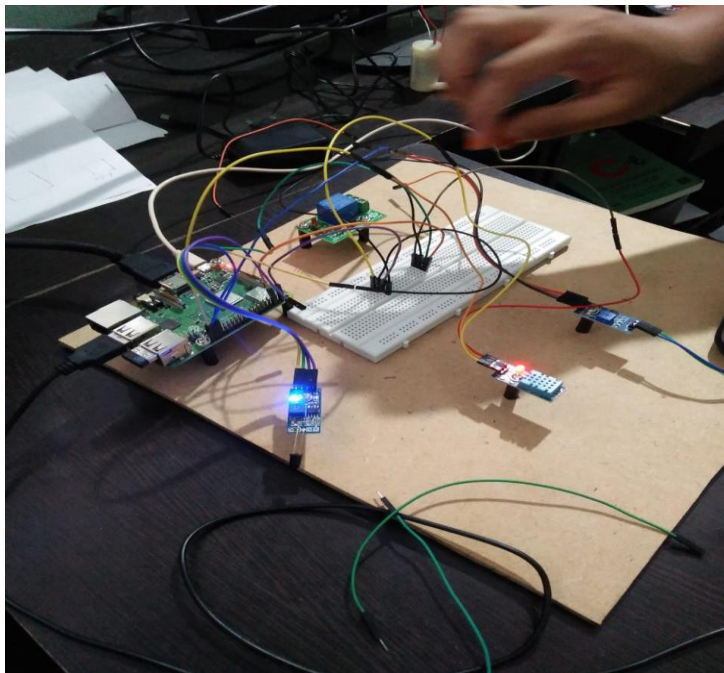
-GND->GND

3)SOIL MOSITURE SENSOR

-VCC->5V

-GND->-GND

-SIG->GPIO16



Appendix : References

- Crop Prediction using Machine Learning Proceedings of the Third International Conference on Smart Systems and Inventive Technology (ICSSIT 2020) IEEE Xplore Part Number: CFP20P17-ART; ISBN: 978-1-7281-5821-1
- Crop Prediction Using Machine Learning Kevin Tom Thomas¹ , Varsha S² , Merin Mary Saji³ , Lisha Varghese⁴ , Er. Jinu Thomas⁵ ^{1,2,3,4} UG students Department Computer Engineering, SAINTGITS College of Engineering, APJ ABDUL KALAM TECHNOLOGICAL University, Kerala, India ⁵Asst. Prof. Department Computer Engineering, SAINTGITS College of Engineering, APJ ABDUL KALAM TECHNOLOGICAL University, Kerala, India
- Wireless Agriculture Monitoring using Raspberry Pi International Journal of Engineering Research & Technology (IJERT) <http://www.ijert.org> ISSN: 2278-0181 IJERTV6IS050217 (This work is licensed under a Creative Commons Attribution 4.0 International License.) Published by : www.ijert.org Vol. 6 Issue 05, May – 2017
- Raspberry pi based real time monitoring of Agriculture & Irrigation Using IOT Athira P. Shaji MTech Student Computer Science and Engineering, School Of Computer Sciences, MG University , Kottayam , india© IJEDR 2018 | Volume 6, Issue 2 | ISSN: 2321-9939
- Internet of Things for Precision Agriculture Applications 2019 Fifth International Conference on Image Information Processing (ICIIP)
- IOT BASED SMART CROP-FIELD MONITORING AND AUTOMATION IRRIGATION SYSTEM Proceedings of the Second International Conference on Inventive Systems and Control (ICISC 2018) IEEE Xplore Compliant - Part Number:CFP18J06-ART, ISBN:978-1-5386-0807-4; DVD Part Number:CFP18J06DVD, ISBN:978-1-5386-0806-7