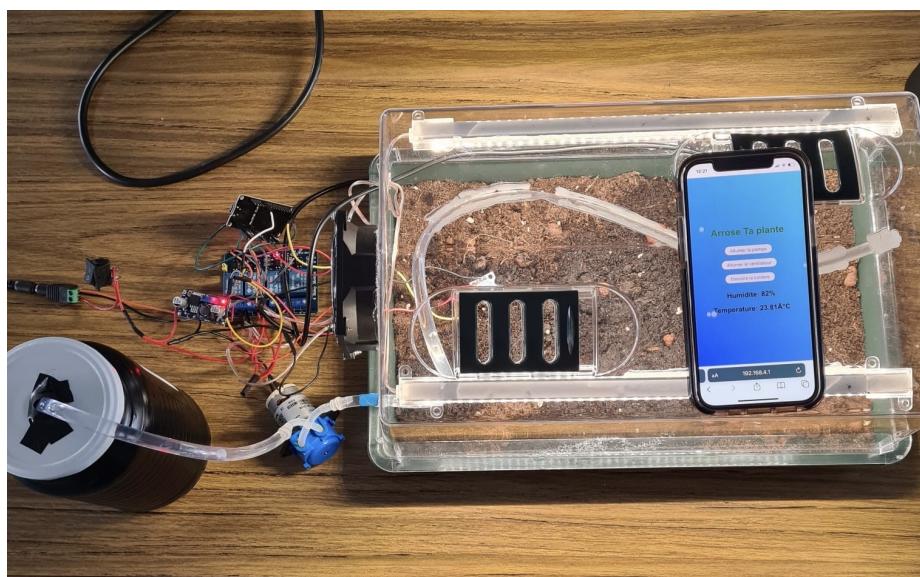


# Serre Conectée

Alec Waumans



## 1 Introduction

Ici nous allons concevoir une serre connectée ou à partir de n'importe quel browser on peut contrôler la serre du moment qu'on est sur le même wifi. Mais de plus, elle sera complètement automatisée. Nous allons nous concentrer du coup sur trois systèmes qui vont s'occuper des plantes. Le premier est un système d'arrosage automatique et contrôlable, un système de lumière automatique et contrôlable, et un système de ventilation automatique et contrôlable.

## 2 Materiel

- Contrôleur : ESP32
  - Pompe péristaltique (12volts)
  - Lumière : bande LED (5volts)
  - Réservoir + tube
  - Alimentation 12 volts
  - 3 relais électriques
  - Ventilateur d'ordinateur (12 volts)
  - Capteur de température (DS18B20)
  - Capteur d'humidité
  - Mini serre en plastique
  - Fer à souder
  - Câble
  - Colle chaude
  - convertisseur 12 volts à 5 volts

3 Code

```

1 #include <WiFi.h>
2 #include <iomanip>
3 #include <WebServer.h>
4
5 #include <OneWire.h>
6 #include <DallasTemperature.h>
7
8 #define ONE_WIRE_BUS 4 // Broche à laquelle le capteur DS18B20 est connecté
9
10 OneWire oneWire(ONE_WIRE_BUS);
11 DallasTemperature sensors(&oneWire);
12
13
14 const char *ssid = "ESP32AP"; // Nom du point d'accès WiFi
15 const char *password = "12345678"; // Mot de passe du point d'accès
16
17 WebServer server(80);
18
19
20 const int relayP1light = 27; // Numéro de la broche GPIO connectée au relais
21 const int relayP1ventilateur = 33; // Numéro de la broche GPIO connectée au ventilateur
22 const int relayP1pompe = 32; // Numéro de la broche GPIO connectée au relais
23
24
25 int sensorPinHumidité = 34; // Numéro de la broche GPIO connectée au capteur d'humidité
26
27 int value = 8;
28 int humidityPercentage = 0;
29
30
31 bool isBumpon = false; // Variable pour suivre l'état de la pompe
32 bool isFanOn = false; // Variable pour suivre l'état du ventilateur
33 bool isLightOn = false; // Variable pour suivre l'état de la lumière
34
35 void setup() {
36   pinMode(relayP1light, OUTPUT);
37   pinMode(relayP1ventilateur, OUTPUT);
38   pinMode(relayP1pompe, OUTPUT);
39
40   digitalWrite(relayP1light, LOW); // Initialisation du relais à l'état désactivé
41   digitalWrite(relayP1ventilateur, LOW); // Initialisation du relais à l'état désactivé
42   digitalWrite(relayP1pompe, LOW); // Initialisation du relais à l'état désactivé
43
44   sensors.begin();
45   delay(1000);
46
47   Serial.begin(9600);
48   delay(500);
49   Serial.println("Initialisation...");
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
101
102
103
104
105
106
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
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
649
650
651
652
653
654
655
656
657
658
659
659
660
661
662
663
664
665
666
667
668
669
669
670
671
672
673
674
675
676
677
678
679
679
680
681
682
683
684
685
686
687
687
688
689
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
709
710
711
712
713
714
715
716
717
718
719
719
720
721
722
723
724
725
726
727
727
728
729
729
730
731
732
733
734
735
736
737
737
738
739
739
740
741
742
743
744
744
745
746
746
747
747
748
748
749
749
750
751
752
753
753
754
755
755
756
756
757
757
758
758
759
759
760
761
761
762
762
763
763
764
764
765
765
766
766
767
767
768
768
769
769
770
770
771
771
772
772
773
773
774
774
775
775
776
776
777
777
778
778
779
779
780
780
781
781
782
782
783
783
784
784
785
785
786
786
787
787
788
788
789
789
790
790
791
791
792
792
793
793
794
794
795
795
796
796
797
797
798
798
799
799
800
800
801
801
802
802
803
803
804
804
805
805
806
806
807
807
808
808
809
809
810
810
811
811
812
812
813
813
814
814
815
815
816
816
817
817
818
818
819
819
820
820
821
821
822
822
823
823
824
824
825
825
826
826
827
827
828
828
829
829
830
830
831
831
832
832
833
833
834
834
835
835
836
836
837
837
838
838
839
839
840
840
841
841
842
842
843
843
844
844
845
845
846
846
847
847
848
848
849
849
850
850
851
851
852
852
853
853
854
854
855
855
856
856
857
857
858
858
859
859
860
860
861
861
862
862
863
863
864
864
865
865
866
866
867
867
868
868
869
869
870
870
871
871
872
872
873
873
874
874
875
875
876
876
877
877
878
878
879
879
880
880
881
881
882
882
883
883
884
884
885
885
886
886
887
887
888
888
889
889
890
890
891
891
892
892
893
893
894
894
895
895
896
896
897
897
898
898
899
899
900
900
901
901
902
902
903
903
904
904
905
905
906
906
907
907
908
908
909
909
910
910
911
911
912
912
913
913
914
914
915
915
916
916
917
917
918
918
919
919
920
920
921
921
922
922
923
923
924
924
925
925
926
926
927
927
928
928
929
929
930
930
931
931
932
932
933
933
934
934
935
935
936
936
937
937
938
938
939
939
940
940
941
941
942
942
943
943
944
944
945
945
946
946
947
947
948
948
949
949
950
950
951
951
952
952
953
953
954
954
955
955
956
956
957
957
958
958
959
959
960
960
961
961
962
962
963
963
964
964
965
965
966
966
967
967
968
968
969
969
970
970
971
971
972
972
973
973
974
974
975
975
976
976
977
977
978
978
979
979
980
980
981
981
982
982
983
983
984
984
985
985
986
986
987
987
988
988
989
989
990
990
991
991
992
992
993
993
994
994
995
995
996
996
997
997
998
998
999
999
1000
1000
1001
1001
1002
1002
1003
1003
1004
1004
1005
1005
1006
1006
1007
1007
1008
1008
1009
1009
1010
1010
1011
1011
1012
1012
1013
1013
1014
1014
1015
1015
1016
1016
1017
1017
1018
1018
1019
1019
1020
1020
1021
1021
1022
1022
1023
1023
1024
1024
1025
1025
1026
1026
1027
1027
1028
1028
1029
1029
1030
1030
1031
1031
1032
1032
1033
1033
1034
1034
1035
1035
1036
1036
1037
1037
1038
1038
1039
1039
1040
1040
1041
1041
1042
1042
1043
1043
1044
1044
1045
1045
1046
1046
1047
1047
1048
1048
1049
1049
1050
1050
1051
1051
1052
1052
1053
1053
1054
1054
1055
1055
1056
1056
1057
1057
1058
1058
1059
1059
1060
1060
1061
1061
1062
1062
1063
1063
1064
1064
1065
1065
1066
1066
1067
1067
1068
1068
1069
1069
1070
1070
1071
1071
1072
1072
1073
1073
1074
1074
1075
1075
1076
1076
1077
1077
1078
1078
1079
1079
1080
1080
1081
1081
1082
1082
1083
1083
1084
1084
1085
1085
1086
1086
1087
1087
1088
1088
1089
1089
1090
1090
1091
1091
1092
1092
1093
1093
1094
1094
1095
1095
1096
1096
1097
1097
1098
1098
1099
1099
1100
1100
1101
1101
1102
1102
1103
1103
1104
1104
1105
1105
1106
1106
1107
1107
1108
1108
1109
1109
1110
1110
1111
1111
1112
1112
1113
1113
1114
1114
1115
1115
1116
1116
1117
1117
1118
1118
1119
1119
1120
1120
1121
1121
1122
1122
1123
1123
1124
1124
1125
1125
1126
1126
1127
1127
1128
1128
1129
1129
1130
1130
1131
1131
1132
1132
1133
1133
1134
1134
1135
1135
1136
1136
1137
1137
1138
1138
1139
1139
1140
1140
1141
1141
1142
1142
1143
1143
1144
1144
1145
1145
1146
1146
1147
1147
1148
1148
1149
1149
1150
1150
1151
1151
1152
1152
1153
1153
1154
1154
1155
1155
1156
1156
1157
1157
1158
1158
1159
1159
1160
1160
1161
1161
1162
1162
1163
1163
1164
1164
1165
1165
1166
1166
1167
1167
1168
1168
1169
1169
1170
1170
1171
1171
1172
1172
1173
1173
1174
1174
1175
1175
1176
1176
1177
1177
1178
1178
1179
1179
1180
1180
1181
1181
1182
1182
1183
1183
1184
1184
1185
1185
1186
1186
1187
1187
1188
1188
1189
1189
1190
1190
1191
1191
1192
1192
1193
1193
1194
1194
1195
1195
1196
1196
1197
1197
1198
1198
1199
1199
1200
1200
1201
1201
1202
1202
1203
1203
1204
1204
1205
1205
1206
1206
1207
1207
1208
1208
1209
1209
1210
1210
1211
1211
1212
1212
1213
1213
1214
1214
1215
1215
1216
1216
1217
1217
1218
1218
1219
1219
1220
1220
1221
1221
1222
1222
1223
1223
1224
1224
1225
1225
1226
1226
1227
1227
1228
1228
1229
1229
1230
1230
1231
1231
1232
1232
1233
1233
1234
1234
1235
1235
1236
1236
1237
1237
1238
1238
1239
1239
1240
1240
1241
1241
1242
1242
1243
1243
1244
1244
1245
1245
1246
1246
1247
1247
1248
1248
1249
1249
1250
1250
1251
1251
1252
1252
1253
1253
1254
1254
1255
1255
1256
1256
1257
1257
1258
1258
1259
1259
1260
1260
1261
1261
1262
1262
1263
1263
1264
1264
1265
1265
1266
1266
1267
1267
1268
1268
1269
1269
1270
1270
1271
1271
1272
1272
1273
1273
1274
1274
1275
1275
1276
1276
1277
1277
1278
1278
1279
1279
1280
1280
1281
1281
1282
1282
1283
1283
1284
1284
1285
1285
1286
1286
1287
1287
1288
1288
1289
1289
1290
1290
1291
1291
1292
1292
1293
1293
1294
1294
1295
1295
1296
1296
1297
1297
1298
1298
1299
1299
1300
1300
1301
1301
1302
1302
1303
1303
1304
1304
1305
1305
1306
1306
1307
1307
1308
1308
1309
1309
1310
1310
1311
1311
1312
1312
1313
1313
1314
1314
1315
1315
1316
1316
1317
1317
1318
1318
1319
1319
1320
1320
1321
1321
1322
1322
1323
1323
1324
1324
1325
1325
1326
1326
1327
1327
1328
1328
1329
1329
1330
1330
1331
1331
1332
1332
1333
1333
1334
1334
1335
1335
1336
1336
1337
1337
1338
1338
1339
1339
1340
1340
1341
1341
1342
1342
1343
1343
1344
1344
1345
1345
1346
1346
1347
1347
1348
1348
1349
1349
1350
1350
1351
1351
1352
1352
1353
1353
1354
1354
1355
1355
1356
1356
1357
1357
1358
1358
1359
1359
1360
1360
1361
1361
1362
1362
1363
1363
1364
1364
1365
1365
1366
1366
1367
1367
1368
1368
1369
1369
1370
1370
1371
1371
1372
1372
1373
1373
1374
1374
1375
1375
1376
1376
1377
1377
1378
1378
1379
1379
1380
1380
1381
1381
1382
1382
1383
1383
1384
1384
1385
1385
1386
1386
1387
1387
1388
1388
1389
1389
1390
1390
1391
1391
1392
1392
1393
1393
1394
1394
1395
1395
1396
1396
1397
1397
1398
1398
1399
1399
1400
1400
1401
1401
1402
1402
1403
1403
1404
1404
1405
1405
1406
1406
1407
1407
1408
1408
1409
1409
1410
1410
1411
1411
1412
1412
1413
1413
1414
1414
1415
1415
1416
1416
1417
1417
1418
1418
1419
1419
1420
1420
1421
1421
1422
1422
1423
1423
1424
1424
1425
1425
1426
1426
1427
1427
1428
1428
1429
1429
1430
1430
1431
1431
1432
1432
1433
1433
1434
1434
1435
1435
1436
1436
1437
1437
1438
1438
1439
1439
1440
1440
1441
1441
1442
1442
1443
1443
1444
1444
1445
1445
1446
1446
1447
1447
1448
1448
1449
1449
1450
1450
1451
1451
1452
1452
1453
1453
1454
1454
1455
1455
1456
1456
1457
1457
1458
1458
1459
1459
1460
1460
1461
1461
1462
1462
1463
1463
1464
1464
1465
1465
1466
1466
1467
1467
1468
1468
1469
1469
1470
1470
1471
1471
1472
1472
1473
1473
1474
1474
1475
1475
1476
1476
1477
1477
1478
1478
1479
1479
1480
1480
1481
1481
1482
1482
1483
1483
1484
1484
1485
1485
1486
1486
1487
1487
1488
1488
1489
1489
1490
1490
1491
1491
1492
1492
1493
1493
1494
1494
1495
1495
1496
1496
1497
1497
1498
1498
1499
1499
1500
1500
1501
1501
1502
1502
1503
1503
1504
1504
1505
1505
1506
1506
1507
1507
1508
1508
1509
1509
1510
1510
1511
1511
1512
1512
1513
1513
1514
1514
1515
1515
1516
1516
1517
1517
1518
1518
1519
1519
1520
1520
1521
1521
1522
1522
1523
1523
1524
1524
1525
1525
1526
1526
1527
1527
1528
1528
1529
1529
1530
1530
1531
1531
1532
1532
1533
1533
1534
1534
1535
1535
1536
1536
1537
1537
1538
1538
1539
1539
1540
1540
1541
1541
1542
1542
1543
1543
1544
1544
1545
1545
1546
1546
1547
1547
1548
1548
1549
1549
1550
1550
1551
1551
1552
1552
1553
1553
1554
1554
1555
1555
1556
1556
1557
1557
1558
1558
1559
1559
1560
1560
1561
1561
1562
1562
1563
1563
1564
1564
1565
1565
1566
1566
1567
1567
1568
1568
1569
1569
1570
1570
1571
1571
1572
1572
1573
1573
1574
1574
1575
1575
1576
1576
1577
1577
1578
1578
1579
1579
1580
1580
1581
1581
1582
1582
1583
1583
1584
1584
1585
1585
1586
1586
1587
1587
1588
1588
1589
1589
1590
1590
1591
1591
1592
1592
1593
1593
1594
1594
1595
1595
1596
1596
1597
1597
1598
1598
1599
1599
1600
1600
1601
1601
1602
1602
1603
1603
1604
1604
1605
1605
1606
1606
1607
1607
1608
1608
1609
1609
1610
1610
1611
1611
1612
1612
1613
1613
1614
1614
1615
1615
1616
1616
1617
1617
1618
1618
1619
1619
1620
1620
1621
1621
1622
1622
1623
1623
1624
1624
1625
1625
1626
1626
1627
1627
1628
1628
1629
1629
1630
1630
1631
1631
1632
1632
1633
1633
1634
1634
1635
1635
1636
1636
1637
1637
1638
1638
1639
1639
1640
1640
1641
1641
1642
1642
1643
1643
1644
1644
1645
1645
1646
1646
1647
1647
1648
1648
1649
1649
1650
1650
1651
1651
1652
1652
1653
1653
1654
1654
1655
1655
1656
1656
1657
1657
1658
1658
1659
1659
1660
1660
1661
1661
1662
1662
1663
1663
1664
1664
1665
1665
1666
1666
1667
1667
1668
1668
1669
1669
1670
1670
1671
1671
1672
1672
1673
1673
1674
1674
1675
1675
1676
1676
1677
1677
1678
1678
1679
1679
1680
1680
1681
1681
1682
1682
1683
1683
1684
1684
1685
1685
1686
1686
1687
1687
1688
1688
1689
1689
1690
1690
1691
1691
1692
1692
1693
1693
1694
1694
1695
1695
1696
1696
1
```

Figure 1: Première partie

Figure 2: Deuxième partie

```
1 // This file is part of the Tarsos-DSP Java bindings.
2 // See the file "LICENSE" for more information.
3 
4 package com.tarsos.dsp;
5 
6 import java.util.ArrayList;
7 import java.util.List;
8 
9 import org.jtransforms.FFT;
10 import org.jtransforms.FFTWindow;
11 import org.jtransforms.Window;
12 
13 /**
14  * A class that provides a windowed version of the Discrete Fourier Transform
15  * (DFT). The DFT is a linear operation that takes a signal and produces its
16  * frequency spectrum. This class provides a windowed version of the DFT, which
17  * is useful for analyzing signals that are not stationary or have a non-zero
18  * mean. The windowed DFT is computed by multiplying the signal by a window
19  * function before applying the DFT. This results in a more accurate estimate
20  * of the signal's frequency spectrum.
21  */
22 public class WindowedDFT {
23 
24     private static final int MAX_N = 1024;
25 
26     private static final double PI = 3.141592653589793;
27 
28     private static final double TWO_PI = 6.283185307179586;
29 
30     private static final double SQRT_PI = 1.778279410038923;
31 
32     private static final double SQRT_2PI = 2.506628965443124;
33 
34     private static final double SQRT_3PI = 3.678794411386332;
35 
36     private static final double SQRT_4PI = 4.898979485566356;
37 
38     private static final double SQRT_5PI = 6.148513513527126;
39 
40     private static final double SQRT_6PI = 7.403124237432393;
41 
42     private static final double SQRT_7PI = 8.66233482468029;
43 
44     private static final double SQRT_8PI = 9.899594936580783;
45 
46     private static final double SQRT_9PI = 11.110674005796008;
47 
48     private static final double SQRT_10PI = 12.309677068735136;
49 
50     private static final double SQRT_11PI = 13.49065837644823;
51 
52     private static final double SQRT_12PI = 14.655850252819308;
53 
54     private static final double SQRT_13PI = 15.79983784537687;
55 
56     private static final double SQRT_14PI = 16.92743640687452;
57 
58     private static final double SQRT_15PI = 18.04779714613529;
59 
60     private static final double SQRT_16PI = 19.16087314724187;
61 
62     private static final double SQRT_17PI = 20.26667346093757;
63 
64     private static final double SQRT_18PI = 21.36520598931977;
65 
66     private static final double SQRT_19PI = 22.45647187563277;
67 
68     private static final double SQRT_20PI = 23.54054887756627;
69 
70     private static final double SQRT_21PI = 24.61747785398127;
71 
72     private static final double SQRT_22PI = 25.68725774779227;
73 
74     private static final double SQRT_23PI = 26.750000000000004;
75 
76     private static final double SQRT_24PI = 27.80562314579638;
77 
78     private static final double SQRT_25PI = 28.85419737956314;
79 
80     private static final double SQRT_26PI = 29.89569245743294;
81 
82     private static final double SQRT_27PI = 30.929999999999995;
83 
84     private static final double SQRT_28PI = 31.95619642384747;
85 
86     private static final double SQRT_29PI = 32.97529931690447;
87 
88     private static final double SQRT_30PI = 33.98729931690447;
89 
90     private static final double SQRT_31PI = 34.99209893887307;
91 
92     private static final double SQRT_32PI = 35.99069931690447;
93 
94     private static final double SQRT_33PI = 36.98259893887307;
95 
96     private static final double SQRT_34PI = 37.97749931690447;
97 
98     private static final double SQRT_35PI = 38.96539893887307;
99 
100    private static final double SQRT_36PI = 39.95529931690447;
101 
102    private static final double SQRT_37PI = 40.94719893887307;
103 
104    private static final double SQRT_38PI = 41.93119931690447;
105 
106    private static final double SQRT_39PI = 42.91719893887307;
107 
108    private static final double SQRT_40PI = 43.90519931690447;
109 
110    private static final double SQRT_41PI = 44.89519893887307;
111 
112    private static final double SQRT_42PI = 45.88719931690447;
113 
114    private static final double SQRT_43PI = 46.87119893887307;
115 
116    private static final double SQRT_44PI = 47.85719931690447;
117 
118    private static final double SQRT_45PI = 48.84519893887307;
119 
120    private static final double SQRT_46PI = 49.83419931690447;
121 
122    private static final double SQRT_47PI = 50.82419893887307;
123 
124    private static final double SQRT_48PI = 51.81519931690447;
125 
126    private static final double SQRT_49PI = 52.80719893887307;
127 
128    private static final double SQRT_50PI = 53.79919931690447;
129 
130    private static final double SQRT_51PI = 54.79219893887307;
131 
132    private static final double SQRT_52PI = 55.78619931690447;
133 
134    private static final double SQRT_53PI = 56.78019893887307;
135 
136    private static final double SQRT_54PI = 57.77419931690447;
137 
138    private static final double SQRT_55PI = 58.76819893887307;
139 
140    private static final double SQRT_56PI = 59.76219931690447;
141 
142    private static final double SQRT_57PI = 60.75619893887307;
143 
144    private static final double SQRT_58PI = 61.75019931690447;
145 
146    private static final double SQRT_59PI = 62.74419893887307;
147 
148    private static final double SQRT_60PI = 63.73819931690447;
149 
150    private static final double SQRT_61PI = 64.73219893887307;
151 
152    private static final double SQRT_62PI = 65.72619931690447;
153 
154    private static final double SQRT_63PI = 66.72019893887307;
155 
156    private static final double SQRT_64PI = 67.71419931690447;
157 
158    private static final double SQRT_65PI = 68.70819893887307;
159 
160    private static final double SQRT_66PI = 69.70219931690447;
161 
162    private static final double SQRT_67PI = 70.69619893887307;
163 
164    private static final double SQRT_68PI = 71.69019931690447;
165 
166    private static final double SQRT_69PI = 72.68419893887307;
167 
168    private static final double SQRT_70PI = 73.67819931690447;
169 
170    private static final double SQRT_71PI = 74.67219893887307;
171 
172    private static final double SQRT_72PI = 75.66619931690447;
173 
174    private static final double SQRT_73PI = 76.66019893887307;
175 
176    private static final double SQRT_74PI = 77.65419931690447;
177 
178    private static final double SQRT_75PI = 78.64819893887307;
179 
180    private static final double SQRT_76PI = 79.64219931690447;
181 
182    private static final double SQRT_77PI = 80.63619893887307;
183 
184    private static final double SQRT_78PI = 81.63019931690447;
185 
186    private static final double SQRT_79PI = 82.62419893887307;
187 
188    private static final double SQRT_80PI = 83.61819931690447;
189 
190    private static final double SQRT_81PI = 84.61219893887307;
191 
192    private static final double SQRT_82PI = 85.60619931690447;
193 
194    private static final double SQRT_83PI = 86.60019893887307;
195 
196    private static final double SQRT_84PI = 87.59419931690447;
197 
198    private static final double SQRT_85PI = 88.58819893887307;
199 
200    private static final double SQRT_86PI = 89.58219931690447;
201 
202    private static final double SQRT_87PI = 90.57619893887307;
203 
204    private static final double SQRT_88PI = 91.57019931690447;
205 
206    private static final double SQRT_89PI = 92.56419893887307;
207 
208    private static final double SQRT_90PI = 93.55819931690447;
209 
210    private static final double SQRT_91PI = 94.55219893887307;
211 
212    private static final double SQRT_92PI = 95.54619931690447;
213 
214    private static final double SQRT_93PI = 96.54019893887307;
215 
216    private static final double SQRT_94PI = 97.53419931690447;
217 
218    private static final double SQRT_95PI = 98.52819893887307;
219 
220    private static final double SQRT_96PI = 99.52219931690447;
221 
222    private static final double SQRT_97PI = 100.51619893887307;
223 
224    private static final double SQRT_98PI = 101.51019931690447;
225 
226    private static final double SQRT_99PI = 102.50419893887307;
227 
228    private static final double SQRT_100PI = 103.50019931690447;
229 
230    private static final double SQRT_101PI = 104.49419893887307;
231 
232    private static final double SQRT_102PI = 105.48819931690447;
233 
234    private static final double SQRT_103PI = 106.48219893887307;
235 
236    private static final double SQRT_104PI = 107.47619931690447;
237 
238    private static final double SQRT_105PI = 108.47019893887307;
239 
240    private static final double SQRT_106PI = 109.46419931690447;
241 
242    private static final double SQRT_107PI = 110.45819893887307;
243 
244    private static final double SQRT_108PI = 111.45219931690447;
245 
246    private static final double SQRT_109PI = 112.44619893887307;
247 
248    private static final double SQRT_110PI = 113.44019931690447;
249 
250    private static final double SQRT_111PI = 114.43419893887307;
251 
252    private static final double SQRT_112PI = 115.42819931690447;
253 
254    private static final double SQRT_113PI = 116.42219893887307;
255 
256    private static final double SQRT_114PI = 117.41619931690447;
257 
258    private static final double SQRT_115PI = 118.41019893887307;
259 
260    private static final double SQRT_116PI = 119.40419931690447;
261 
262    private static final double SQRT_117PI = 120.39819893887307;
263 
264    private static final double SQRT_118PI = 121.39219931690447;
265 
266    private static final double SQRT_119PI = 122.38619893887307;
267 
268    private static final double SQRT_120PI = 123.38019931690447;
269 
270    private static final double SQRT_121PI = 124.37419893887307;
271 
272    private static final double SQRT_122PI = 125.36819931690447;
273 
274    private static final double SQRT_123PI = 126.36219893887307;
275 
276    private static final double SQRT_124PI = 127.35619931690447;
277 
278    private static final double SQRT_125PI = 128.35019893887307;
279 
280    private static final double SQRT_126PI = 129.34419931690447;
281 
282    private static final double SQRT_127PI = 130.33819893887307;
283 
284    private static final double SQRT_128PI = 131.33219931690447;
285 
286    private static final double SQRT_129PI = 132.32619893887307;
287 
288    private static final double SQRT_130PI = 133.32019931690447;
289 
290    private static final double SQRT_131PI = 134.31419893887307;
291 
292    private static final double SQRT_132PI = 135.30819931690447;
293 
294    private static final double SQRT_133PI = 136.30219893887307;
295 
296    private static final double SQRT_134PI = 137.29619931690447;
297 
298    private static final double SQRT_135PI = 138.29019893887307;
299 
300    private static final double SQRT_136PI = 139.28419931690447;
301 
302    private static final double SQRT_137PI = 140.27819893887307;
303 
304    private static final double SQRT_138PI = 141.27219931690447;
305 
306    private static final double SQRT_139PI = 142.26619893887307;
307 
308    private static final double SQRT_140PI = 143.26019931690447;
309 
310    private static final double SQRT_141PI = 144.25419893887307;
311 
312    private static final double SQRT_142PI = 145.24819931690447;
313 
314    private static final double SQRT_143PI = 146.24219893887307;
315 
316    private static final double SQRT_144PI = 147.23619931690447;
317 
318    private static final double SQRT_145PI = 148.23019893887307;
319 
320    private static final double SQRT_146PI = 149.22419931690447;
321 
322    private static final double SQRT_147PI = 150.21819893887307;
323 
324    private static final double SQRT_148PI = 151.21219931690447;
325 
326    private static final double SQRT_149PI = 152.20619893887307;
327 
328    private static final double SQRT_150PI = 153.20019931690447;
329 
330    private static final double SQRT_151PI = 154.19419893887307;
331 
332    private static final double SQRT_152PI = 155.18819931690447;
333 
334    private static final double SQRT_153PI = 156.18219893887307;
335 
336    private static final double SQRT_154PI = 157.17619931690447;
337 
338    private static final double SQRT_155PI = 158.17019893887307;
339 
340    private static final double SQRT_156PI = 159.16419931690447;
341 
342    private static final double SQRT_157PI = 160.15819893887307;
343 
344    private static final double SQRT_158PI = 161.15219931690447;
345 
346    private static final double SQRT_159PI = 162.14619893887307;
347 
348    private static final double SQRT_160PI = 163.14019931690447;
349 
350    private static final double SQRT_161PI = 164.13419893887307;
351 
352    private static final double SQRT_162PI = 165.12819931690447;
353 
354    private static final double SQRT_163PI = 166.12219893887307;
355 
356    private static final double SQRT_164PI = 167.11619931690447;
357 
358    private static final double SQRT_165PI = 168.11019893887307;
359 
360    private static final double SQRT_166PI = 169.10419931690447;
361 
362    private static final double SQRT_167PI = 170.09819893887307;
363 
364    private static final double SQRT_168PI = 171.09219931690447;
365 
366    private static final double SQRT_169PI = 172.08619893887307;
367 
368    private static final double SQRT_170PI = 173.08019931690447;
369 
370    private static final double SQRT_171PI = 174.07419893887307;
371 
372    private static final double SQRT_172PI = 175.06819931690447;
373 
374    private static final double SQRT_173PI = 176.06219893887307;
375 
376    private static final double SQRT_174PI = 177.05619931690447;
377 
378    private static final double SQRT_175PI = 178.05019893887307;
379 
380    private static final double SQRT_176PI = 179.04419931690447;
381 
382    private static final double SQRT_177PI = 180.03819893887307;
383 
384    private static final double SQRT_178PI = 181.03219931690447;
385 
386    private static final double SQRT_179PI = 182.02619893887307;
387 
388    private static final double SQRT_180PI = 183.02019931690447;
389 
390    private static final double SQRT_181PI = 184.01419893887307;
391 
392    private static final double SQRT_182PI = 185.00819931690447;
393 
394    private static final double SQRT_183PI = 186.00219893887307;
395 
396    private static final double SQRT_184PI = 187.00019931690447;
397 
398    private static final double SQRT_185PI = 188.00019893887307;
399 
400    private static final double SQRT_186PI = 189.00019931690447;
401 
402    private static final double SQRT_187PI = 190.00019893887307;
403 
404    private static final double SQRT_188PI = 191.00019931690447;
405 
406    private static final double SQRT_189PI = 192.00019893887307;
407 
408    private static final double SQRT_190PI = 193.00019931690447;
409 
410    private static final double SQRT_191PI = 194.00019893887307;
411 
412    private static final double SQRT_192PI = 195.00019931690447;
413 
414    private static final double SQRT_193PI = 196.00019893887307;
415 
416    private static final double SQRT_194PI = 197.00019931690447;
417 
418    private static final double SQRT_195PI = 198.00019893887307;
419 
420    private static final double SQRT_196PI = 199.00019931690447;
421 
422    private static final double SQRT_197PI = 200.00019893887307;
423 
424    private static final double SQRT_198PI = 201.00019931690447;
425 
426    private static final double SQRT_199PI = 202.00019893887307;
427 
428    private static final double SQRT_200PI = 203.00019931690447;
429 
430    private static final double SQRT_201PI = 204.00019893887307;
431 
432    private static final double SQRT_202PI = 205.00019931690447;
433 
434    private static final double SQRT_203PI = 206.00019893887307;
435 
436    private static final double SQRT_204PI = 207.00019931690447;
437 
438    private static final double SQRT_205PI = 208.00019893887307;
439 
440    private static final double SQRT_206PI = 209.00019931690447;
441 
442    private static final double SQRT_207PI = 210.00019893887307;
443 
444    private static final double SQRT_208PI = 211.00019931690447;
445 
446    private static final double SQRT_209PI = 212.00019893887307;
447 
448    private static final double SQRT_210PI = 213.00019931690447;
449 
450    private static final double SQRT_211PI = 214.00019893887307;
451 
452    private static final double SQRT_212PI = 215.00019931690447;
453 
454    private static final double SQRT_213PI = 216.00019893887307;
455 
456    private static final double SQRT_214PI = 217.00019931690447;
457 
458    private static final double SQRT_215PI = 218.00019893887307;
459 
460    private static final double SQRT_216PI = 219.00019931690447;
461 
462    private static final double SQRT_217PI = 220.00019893887307;
463 
464    private static final double SQRT_218PI = 221.00019931690447;
465 
466    private static final double SQRT_219PI = 222.00019893887307;
467 
468    private static final double SQRT_220PI = 223.00019931690447;
469 
470    private static final double SQRT_221PI = 224.00019893887307;
471 
472    private static final double SQRT_222PI = 225.00019931690447;
473 
474    private static final double SQRT_223PI = 226.00019893887307;
475 
476    private static final double SQRT_224PI = 227.00019931690447;
477 
478    private static final double SQRT_225PI = 228.00019893887307;
479 
480    private static final double SQRT_226PI = 229.00019931690447;
481 
482    private static final double SQRT_227PI = 230.00019893887307;
483 
484    private static final double SQRT_228PI = 231.00019931690447;
485 
486    private static final double SQRT_229PI = 232.00019893887307;
487 
488    private static final double SQRT_230PI = 233.00019931690447;
489 
490    private static final double SQRT_231PI = 234.00019893887307;
491 
492    private static final double SQRT_232PI = 235.00019931690447;
493 
494    private static final double SQRT_233PI = 236.00019893887307;
495 
496    private static final double SQRT_234PI = 237.00019931690447;
497 
498    private static final double SQRT_235PI = 238.00019893887307;
499 
500    private static final double SQRT_236PI = 239.00019931690447;
501 
502    private static final double SQRT_237PI = 240.00019893887307;
503 
504    private static final double SQRT_238PI = 241.00019931690447;
505 
506    private static final double SQRT_239PI = 242.00019893887307;
507 
508    private static final double SQRT_240PI = 243.00019931690447;
509 
510    private static final double SQRT_241PI = 244.00019893887307;
511 
512    private static final double SQRT_242PI = 245.00019931690447;
513 
514    private static final double SQRT_243PI = 246.00019893887307;
515 
516    private static final double SQRT_244PI = 247.00019931690447;
517 
518    private static final double SQRT_245PI = 248.00019893887307;
519 
520    private static final double SQRT_246PI = 249.00019931690447;
521 
522    private static final double SQRT_247PI = 250.00019893887307;
523 
524    private static final double SQRT_248PI = 251.00019931690447;
525 
526    private static final double SQRT_249PI = 252.00019893887307;
527 
528    private static final double SQRT_250PI = 253.00019931690447;
529 
530    private static final double SQRT_251PI = 254.00019893887307;
531 
532    private static final double SQRT_252PI = 255.00019931690447;
533 
534    private static final double SQRT_253PI = 256.00019893887307;
535 
536    private static final double SQRT_254PI = 257.00019931690447;
537 
538    private static final double SQRT_255PI = 258.00019893887307;
539 
540    private static final double SQRT_256PI = 259.00019931690447;
541 
542    private static final double SQRT_257PI = 260.00019893887307;
543 
544    private static final double SQRT_258PI = 261.00019931690447;
545 
546    private static final double SQRT_259PI = 262.00019893887307;
547 
548    private static final double SQRT_260PI = 263.00019931690447;
549 
550    private static final double SQRT_261PI = 264.00019893887307;
551 
552    private static final double SQRT_262PI = 265.00019931690447;
553 
554    private static final double SQRT_263PI = 266.00019893887307;
555 
556    private static final double SQRT_264PI = 267.00019931690447;
557 
558    private static final double SQRT_265PI = 268.00019893887307;
559 
560    private static final double SQRT_266PI = 269.00019931690447;
561 
562    private static final double SQRT_267PI = 270.00019893887307;
563 
564    private static final double SQRT_268PI = 271.00019931690447;
565 
566    private static final double SQRT_269PI = 272.00019893887307;
567 
568    private static final double SQRT_270PI = 273.00019931690447;
569 
570    private static final double SQRT_271PI = 274.00019893887307;
571 
572    private static final double SQRT_272PI = 275.00019931690447;
573 
574    private static final double SQRT_273PI = 276.00019893887307;
575 
576    private static final double SQRT_274PI = 277.00019931690447;
577 
578    private static final double SQRT_275PI = 278.00019893887307;
579 
580    private static final double SQRT_276PI = 279.00019931690447;
581 
582    private static final double SQRT_277PI = 280.00019893887307;
583 
584    private static final double SQRT_278PI = 281.00019931690447;
585 
586    private static final double SQRT_279PI = 282.00019893887307;
587 
588    private static final double SQRT_280PI = 283.00019931690447;
589 
590    private static final double SQRT_281PI = 284.00019893887307;
591 
592    private static final double SQRT_282PI = 285.00019931690447;
593 
594    private static final double SQRT_283PI = 286.00019893887307;
595 
596    private static final double SQRT_284PI = 287.00019931690447;
597 
598    private static final double SQRT_285PI = 288.00019893887307;
599 
600    private static final double SQRT_286PI = 289.00019931690447;
601 
602    private static final double SQRT_287PI = 290.00019893887307;
603 
604    private static final double SQRT_288PI = 291.00019931690447;
605 
606    private static final double SQRT_289PI = 292.00019893887307;
607 
608    private static final double SQRT_290PI = 293.00019931690447;
609 
610    private static final double SQRT_291PI = 294.00019893887307;
611 
612    private static final double SQRT_292PI = 295.00019931690447;
613 
614    private static final double SQRT_293PI = 296.00019893887307;
615 
616    private static final double SQRT_294PI = 297.00019931690447;
617 
618    private static final double SQRT_295PI = 298.00019893887307;
619 
620    private static final double SQRT_296PI = 299.00019931690447;
621 
622    private static final double SQRT_297PI = 300.00019893887307;
623 
624    private static final double SQRT_298PI = 301.00019931690447;
625 
626    private static final double SQRT_299PI = 302.00019893887307;
627 
628    private static final double SQRT_300PI = 303.00019931690447;
629 
630    private static final double SQRT_301PI = 304.00019893887307;
631 
632    private static final double SQRT_302PI = 305.00019931690447;
633 
634    private static final double SQRT_303PI = 306.00019893887307;
635 
636    private static final double SQRT_304PI = 307.00019931690447;
637 
638    private static final double SQRT_305PI = 308.00019893887307;
639 
640    private static final double SQRT_306PI = 309.00019931690447;
641 
642    private static final double SQRT_307PI = 310.00019893887307;
643 
644    private static final double SQRT_308PI = 311.000199316
```

Figure 4: Quatrième partie

Figure 3: Troisième partie

```

165
166    stat = "success";
167    stat = "function logStatus(stat) {"
168    stat += "var status = new XMLHttpRequest();"
169    stat += "status.onreadystatechange = function() {"
170    stat += "if (this.readyState == 4 & this.status == 200) {"
171    stat += "document.getElementById('humidity').innerHTML = humidity; + ' '+this.responseText + ' '";"
172    stat += "}"
173    stat += "};"
174    stat += "status.open('GET', '/"+path+state+stat+');"
175    stat += "status.send();"
176    stat += "};"
177    stat += "function updateHumidity() {"
178    stat += "var humidity = new XMLHttpRequest();"
179    stat += "humidity.onreadystatechange = function() {"
180    stat += "if (this.readyState == 4 & this.status == 200) {"
181    stat += "document.getElementsByClassName('humidity')[0].innerHTML = humidity; + ' '+this.responseText + ' '";"
182    stat += "}"
183    stat += "};"
184    stat += "humidity.open('GET', '/humidity');"
185    stat += "humidity.send();"
186    stat += "};"
187    stat += "function updateTemperature() {"
188    stat += "var temperature = new XMLHttpRequest();"
189    stat += "temperature.onreadystatechange = function() {"
190    stat += "if (this.readyState == 4 & this.status == 200) {"
191    stat += "document.getElementsByClassName('temperature')[0].innerHTML = 'Temperature: ' + this.responseText + ' '";"
192    stat += "}"
193    stat += "};"
194    stat += "temperature.open('GET', '/temperature');"
195    stat += "temperature.send();"
196    stat += "};"
197    stat += "setInterval(logStatus, 3000); // Mettre à jour l'humidité toutes les secondes"
198    stat += "setInterval(logUpdateTemperature, 3000); // Mettre à jour la température toutes les secondes"
199    stat += "
```

```

164 // ===== Actionneur =====
165 // ===== pompe =====
166 server.on("onPompe", HTTP_GET, () => {
167   String state = server.arg("state");
168   serial.print("Received pompe state: ");
169   serial.println(state);
170   Serial.print(state);
171
172   if (state == "1") {
173     digitalWrite(relayInPompe, HIGH); // Allume la pompe
174   isPumpOn = true;
175   Serial.println("Pompe allumée!");
176   server.sendHeader("Location", "/");
177   server.send(200);
178 } else {
179   digitalWrite(relayInPompe, LOW); // Eteint la pompe
180   isPumpOn = false;
181   Serial.println("Pompe éteinte!");
182   server.sendHeader("Location", "/");
183   server.send(200);
184 }
185 });
186
187 server.on("fan", HTTP_GET, () => {
188   String state = server.arg("state");
189   serial.print("Received fan states: ");
190   serial.println(state);
191
192   if (state == "1") {
193     digitalWrite(relayInVentilateur, HIGH); // Allume le ventilateur
194   isFanOn = true;
195   Serial.println("Ventilateur allumé!");
196   server.sendHeader("Location", "/");
197   server.send(200);
198 } else {
199   digitalWrite(relayInVentilateur, LOW); // Eteint le ventilateur
200   isFanOn = false;
201   Serial.println("Ventilateur éteint !");
202   server.sendHeader("Location", "/");
203   server.send(200);
204 }
205 });

```

Figure 5: Cinquième partie

Figure 6: Sixieme partie

## 4 Explication du code

Dans les premières lignes on déclare les pins qu'on relie à l'esp32 et on déclare aussi le wifi et le mot de passe du wifi, ainsi que l'état des systèmes automatisés. Au début du setup, on initialise quels sont les pins qui vont envoyer du courant qui est de l'information du changement d'état et on écrit que les relais doit être éteints. Ensuite, on initialise le serveur sur l'adresse Ip (192.168.4.1). Les lignes de code HTML vont servir juste à la mise en forme du serveur et aussi pour l'actualisation du serveur. Server.send va juste envoyer les informations au serveur. A la ligne 167, on va récupérer les informations du serveur et si l'utilisateur a cliqué sur le bouton "on" de la pompe ça envoie un message au

```

708     server.on("client", HTTP.GET, (req, res) => {
709       String state = server.getString("state");
710       Serial.println("Lumière : " + state);
711       if (state == "allume") {
712         digitalWrite(relayLight, HIGH); // Allume la lumière
713         I2C.write(0x00, "lumière allumée");
714         Serial.println("Lumière allumée");
715         server.send(200);
716       } else if (state == "extinction") {
717         digitalWrite(relayLight, LOW); // Éteint la lumière
718         I2C.write(0x00, "lumière éteinte");
719         Serial.println("Lumière éteinte");
720         server.send(200);
721       }
722     });
723   }
724   //-----Centrale-----
725   server.on("/humidity", HTTP.GET, () => {
726     value = analogRead(sensorHumidity);
727     humidityPercentage = map(value, 0, 1023, 0, 100); // Conversion de la valeur analogique en pourcentage
728     humidityPercentage = abs(humidityPercentage);
729     Serial.print("Humidité : ");
730     Serial.print(humidityPercentage);
731     send("humidity=" + String(humidityPercentage)); // Envoyer la valeur d'humidité à la page web
732   });
733   // Ajouter un gestionnaire pour l'URL "/température"
734   server.on("/temp", HTTP.GET, () => {
735     // Lecture de la température par le capteur
736     sensors.requestTemperature(); // Demander au capteur de mesurer la température
737     float tempC = sensors.getTempByIndex(0); // Obtenir la température en degrés Celsius
738     float tempF = tempC * 9 / 5 + 32; // Conversion en Fahrenheit
739     // Envoyer la température à la page web
740     server.send(200, "text/plain", String(tempC));
741   });
742   server.begin();
743   Serial.println("Serveur démarré");
744 }

```

Figure 7: Septième partie

```

745 void loop() {
746   //---Lumière---
747   sensors.requestTemperature(); // Demander au capteur de mesurer la température
748   float tempC = sensors.getTempByIndex(0); // Obtenir la température en degrés Celsius
749   // Afficher la température dans la console
750   Serial.print("Température : ");
751   Serial.print(tempC);
752   Serial.print("°C");
753   // Vérifier si la température est supérieure ou égale à 26 degrés Celsius
754   if (tempC >= 26) {
755     digitalWrite(relayVentilateur, HIGH); // Allumer le ventilateur
756     delay(5000); // Attende 5 secondes
757     digitalWrite(relayVentilateur, LOW); // Éteindre le ventilateur
758   }
759   //---Humidité---
760   value = analogRead(sensorHumidity);
761   humidityPercentage = map(value, 0, 1023, 0, 100); // Conversion de la valeur analogique en pourcentage
762   Serial.print("Humidité : ");
763   Serial.print(humidityPercentage);
764   // Vérifier si l'humidité est égale ou inférieure à 38 %
765   if (humidityPercentage <= 38) {
766     delay(5000); // Attende 5 secondes
767     digitalWrite(relayPompe, HIGH); // Allumer la pompe
768     delay(5000); // Attende 5 secondes
769     digitalWrite(relayPompe, LOW); // Éteindre la pompe
770   }
771   //---Horloge---
772   int currentHour = hour();
773   int currentMinute = minute();
774   int currentSecond = second();
775   // Vérifier si l'heure est 10h15
776   if (currentHour == 10 & currentMinute == 15) {
777     digitalWrite(relayLight, HIGH); // Allumer la lampe
778     delay(5000); // Attende 5 secondes
779     digitalWrite(relayLight, LOW); // Éteindre la lampe
780   }
781 }

```

Figure 8: Huitième partie

relai pour lui dire d'alimenter la pompe. Par la suite, on fait la même chose pour le système de ventilation et de lumière.

A la ligne 229, on va récupérer l'information du capteur d'humidité et de température pour l'envoyer par la suite au serveur pour afficher les valeurs sur la page du serveur.

Explication de la loop() : Premièrement on envoie une requête pour la connexion server-client. On demande au capteur de température et on la transforme en Celsius. La logique qui suit est celle d'automatiser le système de ventilation, disant que si la température est plus élevée que 26 degrés, il allume le ventilateur pendant un laps de temps défini et s'arrête comme la boucle tourne tout le temps et très vite si la température est toujours à 26 degrés, le même processus s'active. Et dans la suite du code, on fait la même logique pour l'humidité. Mais c'est relativement différent pour le système des lumières qui est basé sur l'heure de la journée qui va s'allumer à une certaine heure et rester allumé pendant un laps de temps.

Le problème non résolu de cette boucle est que, pendant que les temps d'activation de ces systèmes, tout le système est en attente, ce qui veut dire qu'il ne fonctionne pas en parallèle, ce qui peut se régler en pesant du multithread dans un code python, mais cela est hors de mes compétences pour le moment. Comme on dit, on en apprend tous les jours.

## 5 Circuit et cablage

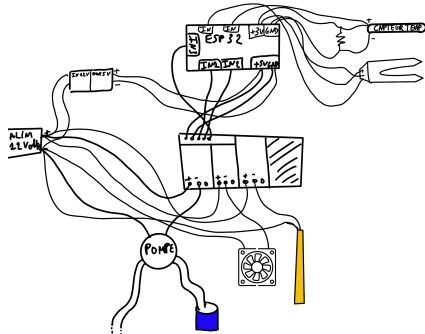


Figure 9: Schémas du circuit

Je n'ai malheureusement pas le projet à côté de moi donc je ne peux être sûr du câblage à 100 pourcent, je vous envoie à regarder les photos en bas du document pour vérifier le câblage.

Première étape connectée les capteurs à l'esp32 En fonction de votre capteur, vous devez vérifier leur alimentation et ainsi connecter les pins de données à des pins analogues de l'esp32 si vous le souhaitez regarder dans mon code pour utiliser les mêmes que les miennes. N'oubliez pas que sur le capteur de température, il faut une résistance entre les deux regardés dans la datasheet pour savoir quelle valeur utiliser.

Alimenter en 5 volts le bloc de relais et connecter les pins d'informations au pin de l'esp32 utilisé les mêmes que les miennes si vous le souhaitez.

Et pour chaque relai connecté, le positif à l'alimentation et le négatif au positif de l'élément à alimenter et le négatif de l'élément au négatif de l'alimentation.

## 6 Amelioration future

Pour la petite boîte qui est censée encadrer les câbles, je mets ci-joint le fichier 3D que vous pouvez directement imprimer via une imprimante 3D.

Il y a ce problème expliqué précédemment que l'on peut régler comme faire un code python avec du multithread qui peut donc par la suite avoir trois systèmes qui peut tourner de façon simultanée.

Il y a le matériel qui peut aussi être mieux adapté avec un peu plus de financement. On peut acheter des lumières horticoles qui poussent à l'amélioration des conditions de vie de la plante. On pourrait adapter le code pour que les temps d'éclairage soient adaptés à certaines plantes. Mais on peut aussi mettre un filtreur d'air avec un filtre carbone pour améliorer la qualité de l'air ou un brumisateur pour les plantes tropicales. Mais pour faire cela, il aurait fallu faire une vraie boîte hermétique. Voici les plans de la boîte.

On pourrait aussi les connecter à Alexa ou Google home ou même faire une serre grande nature la seule limite à ce project est votre imagination, mais aussi votre financement, bien sûr.

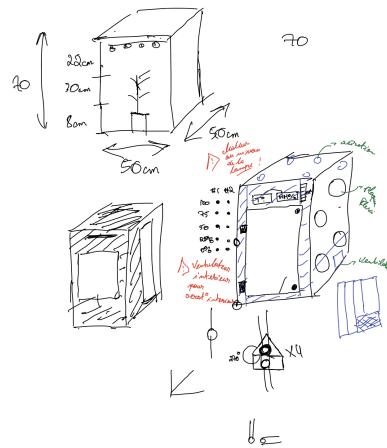


Figure 10: Schémas de la boîte

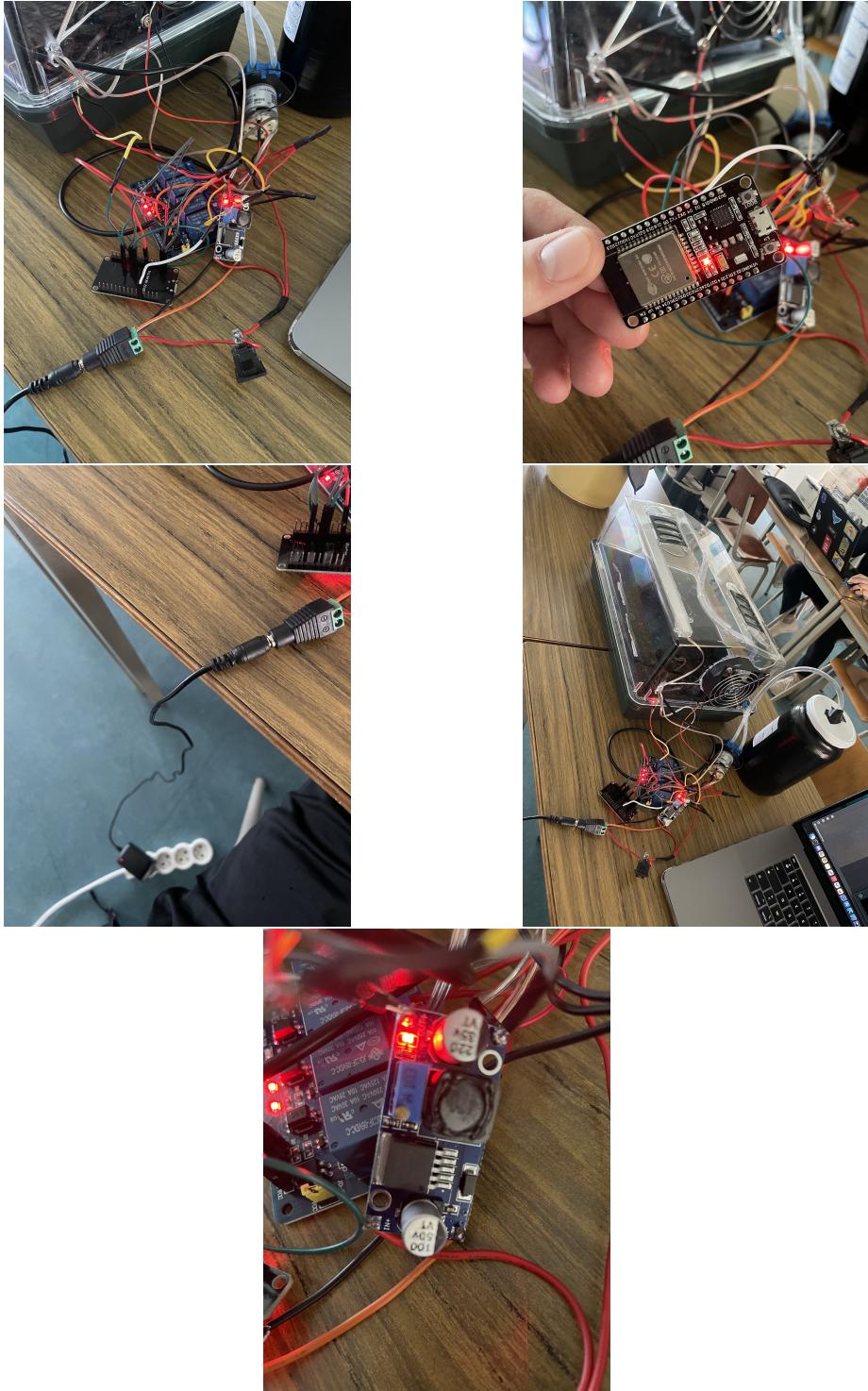


Figure 11: image supplémentaire