5

RESULT AND DISCUSSION

CONTENT

5.1 AUTHENTICATION OF THE SYSTEM FOR USERS 97

5.2 INTEGRATED DATABASE AS COMMON PLATFORM 99

5.3 IVR USER 100

5.3.1 Error Analysis And Speech Recognition Analysis 100

5.4 SPEECH RECOGNITION WITH SPHINX 102

5.4.1 Sphinx Train 102

5.4.2 Sphinx Decode 102

5.4.3 Result Analysis 103

5.5 OUTCOME FOR WI-FI COMMUNICATION 108

5.5.1 Wi-Fi Enabled Circuit Stimulation 108

CHAPTER 5

RESULT AND DISCUSSION

As mentioned in the objective of the proposed system, the aim of this project work is to control electronic or electrical devices which are controlled through a common platform for four communication mechanisms. So, the outcome of this project should be switch on or off these devices in an efficient manner by using a web page, SMS, Speech and Wi-Fi. The result of this system that is outcome finally can be easily visualized with the help of the screen-shots of graphical user interface of software package and hardware modules.

After connecting and switching the all components required in the system administrator comes to the server machine. So, now operation in the server side will begin.

Step 1: At first, run the server program, once if program is executed, then a login interface is appeared in the server machine.

Step 2: Now administrator has to active the server pressing the Start button in the users interface of the server. Similarly, administrator checks the check box for allowing the system to listen remote control command and presses the Connect to GSM modem to listen the commands sent from the GSM end.

After this, the server system is activated and ready to use i.e. ready operate its functions according to the user's request.

Step 3: Now, this time is for Remote users. There are four types of users communicating four different mechanisms as Webpage users, Mobile SMS users, speech user and Wi-Fi User. The web user uses a web page to send the commands using a standard web browser through a dynamic IP address. Similarly, the mobile user uses message service from a simple cell-phone.

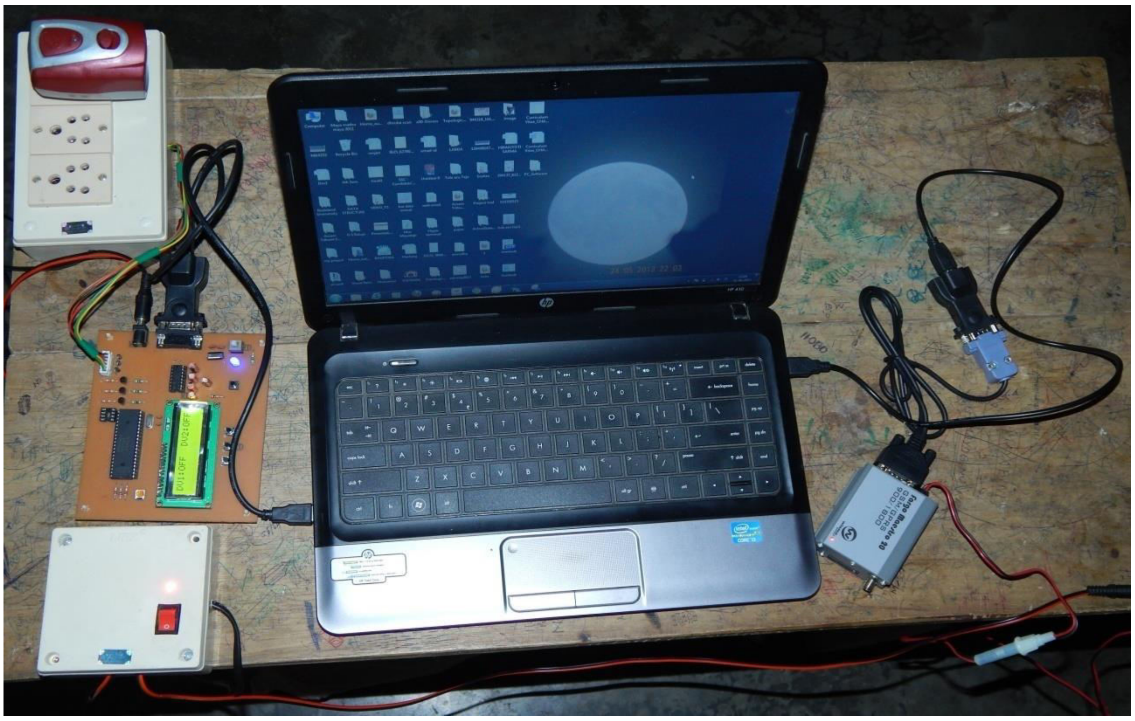
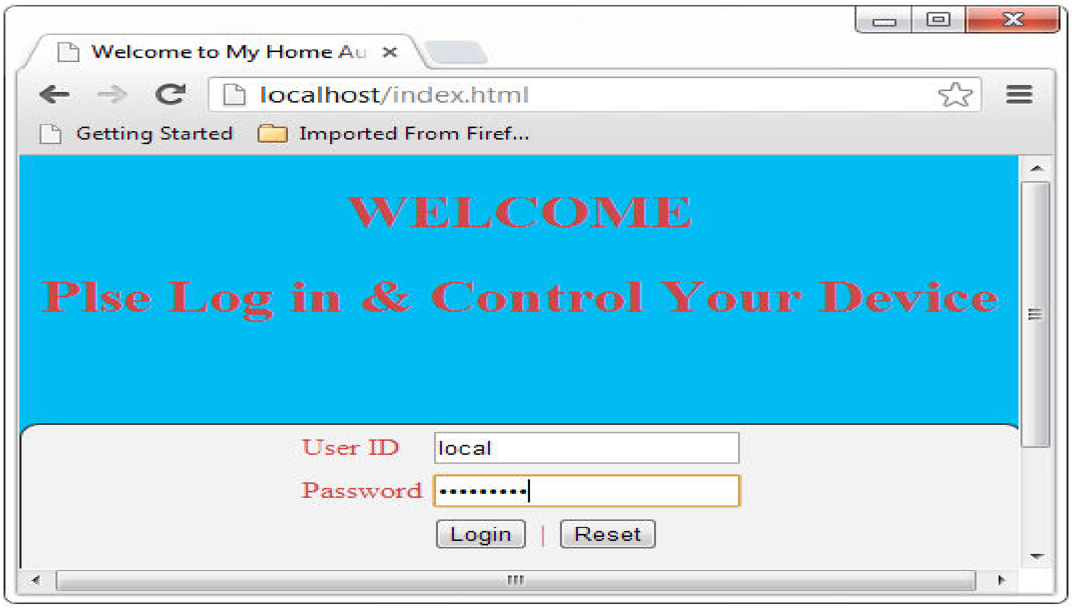


Figure 5.1 Screenshot of one part of present system in server side

5.1 Authentication of the System for Users

In case of the Web and Wifi Communication, remote users have to log in their user_id and password in the user interface as both are interface based in the server end through an IP address.

If user name and password does not match with the database, then the following scenario has been occurred, otherwise can process the system.



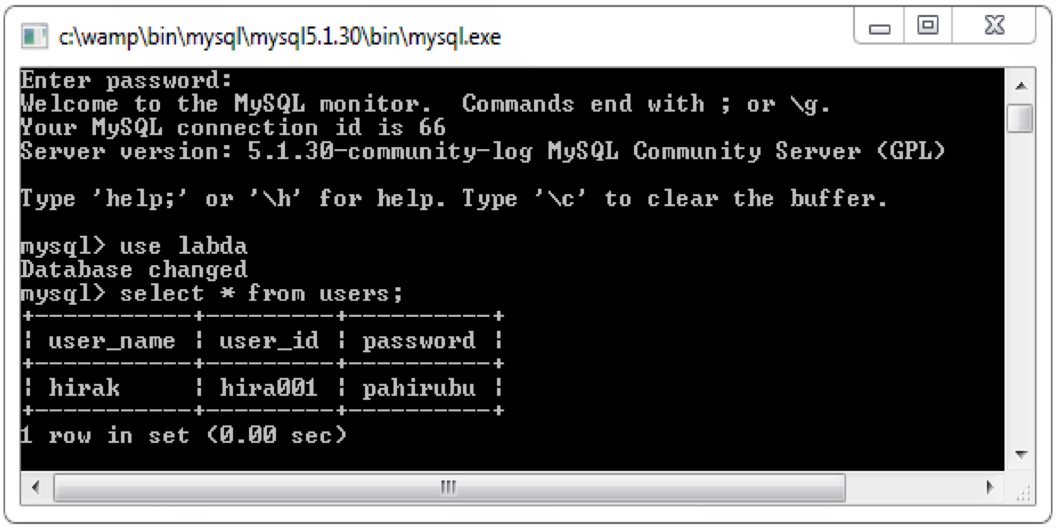
WELCOME

Plse Log in & Control Your Device

User ID local

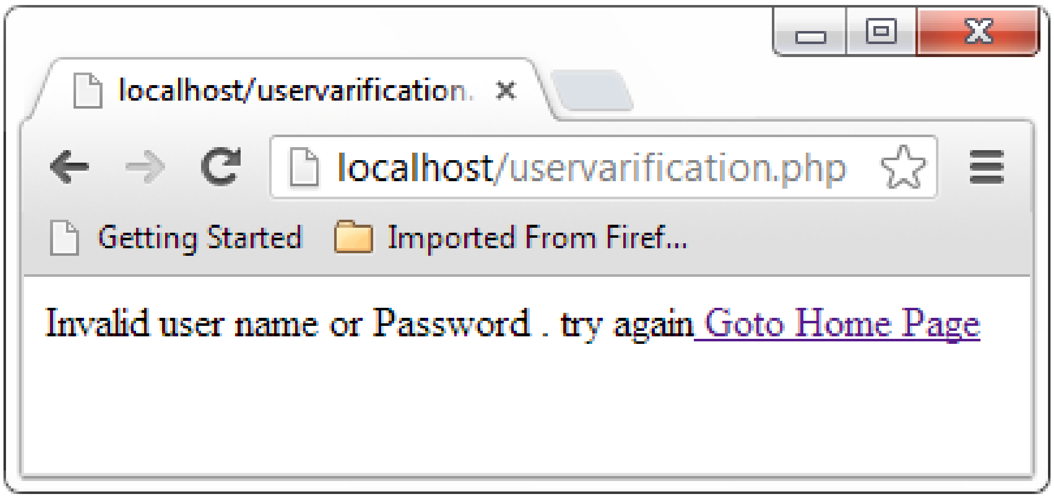
Password ••••••••••

Figure 5.2: User Verification Interface



```
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 66
Server version: 5.1.30-community-log MySQL Community Server (GPL)

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql> use labda
Database changed
mysql> select * from users;
+-----------+----------+----------+
| user_name | user_id  | password |
+-----------+----------+----------+
| hirak     | hira001  | pahirubu |
+-----------+----------+----------+
1 row in set (0.00 sec)
```

Figure 5.3 : Users Information in MySql



Invalid user name or Password . try again Goto Home Page

Figure 5.4: Unauthorized Users

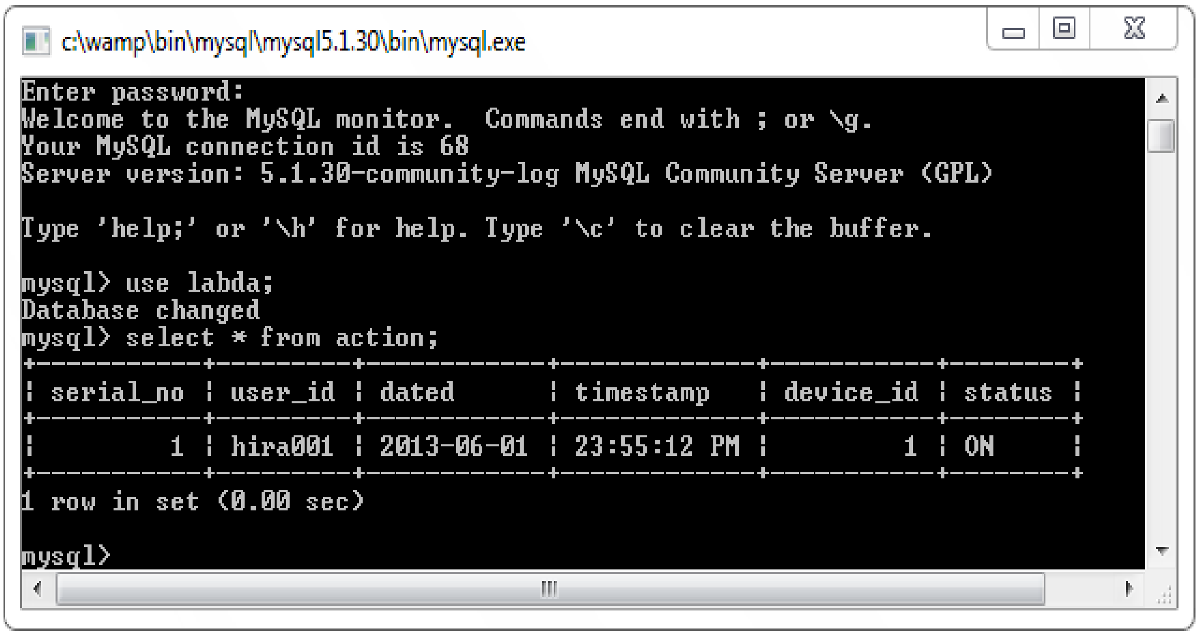
If the user name and password is correct, then the following interface is appeared for controlling the devices.

In case of the SMS Communication, remote users have to send their user_id and password along with the SMS that is already predefined. Besides, the SMS should be sent from an authorized mobile number that is also stored in software package available in the server. Otherwise users will be denied to be executed.

In case of Speech Communication, remote users have to call to the server mobile or GSM Modem to deliver their voice commands to control the devices using the authorized mobile number. Otherwise speech cannot be accepted.

5.2 Integrated Database as Common Platform

The integrated database of the system is the heart of the home automation system because every command actions delivered by different communication mechanisms will be stored in a same database to control the household appliances. The database is responsible for storing the various information about the action commands given by different users. The actual command will be triggered to the microcontroller from the same database as database can carry the same status for the devices.



```
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 68
Server version: 5.1.30-community-log MySQL Community Server (GPL)

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql> use labda;
Database changed
mysql> select * from action;
+-----------+---------+------------+-------------+-----------+--------+
| serial_no | user_id | dated      | timestamp   | device_id | status |
+-----------+---------+------------+-------------+-----------+--------+
|         1 | hira001 | 2013-06-01 | 23:55:12 PM |         1 | ON     |
+-----------+---------+------------+-------------+-----------+--------+
1 row in set (0.00 sec)

mysql>
```

Figure 5.5: action table in MySql database

In the discussion section, it is claimed that the command reached from all four communication mechanism is stored in the same server database and the feedback to both user is delivered from that database. So, it will possible to get the recent status for both users i.e. if device is switched off by the web user, then also that feedback will be delivered into the GSM user and vice-versa. Again, it will possible to get the recent status for both users i.e. if device is switched off by the speech user, then also that feedback will be delivered into the Wi-Fi users and vice-versa.

5.3 IVR User

In this system, database consists of 8 different words LIGHT, FAN, FREEZE, TV, MOTOR,YES,NO,ON, OFF and STATUS. Our speech recognition systems consist of total 1000 utterance words taken from different speaker. Including these words we have taken 880 utterance words for training, which are spoken by 200 different users and took them as a trainee in training phase by recognition toolkit. After completion of their training we have tested by new utterance words by the new input different speakers. We took 120 new utterance words from new speakers. After testing phase is completed, we have compared the training and testing phase. Then we have recognized different kinds of sounds as mention below:

1. Matching sound: These are the sounds used in the training model which match with the testing sounds.

2. Non matching sound: These are the sounds used in the training model which do not match with the testing sounds.

3. Silence sound: These are the sounds used in the training which do not show any outcome.

5.3.1 Error Analysis and Speech Recognition Accuracy

Word error rate often referred to as WER is a way to measure the performance of an automatic speech recognition (ASR) system. It is tricky to measure because the “ASR result” can have a different length than the “Voice Input”. The calculated results are shown below in the table.

Table 5.1: List of training word, occurrences in training set and % of accuracy

Name of the Word	Total No. of Words	Recognzed Word (N)	Deletions (D)	Substitutins (S)	Insertions (I)	Mis Recognized	Errors	Word Correction Rate(PC)	Word Accuracy Rate(PA)	Word Error Rate
LIGHT	237	232	0	5	3	5	8	97.84	96.55	3.45
FAN	235	227	1	7	2	7	10	96.48	95.59	4.41
FREEZE	238	228	5	5	1	5	11	95.61	95.18	4.82
AC	235	230	3	2	0	2	5	97.83	97.83	2.17
MOTOR	240	233	2	5	2	5	9	96.99	96.14	3.86
ON	242	238	0	4	3	4	7	98.32	97.06	2.94
OFF	242	230	4	8	4	8	16	94.78	93.04	6.96
YES	239	232	3	4	2	4	9	96.98	96.12	3.88
NO	238	232	2	4	0	4	6	97.41	97.41	2.59

The system performances in terms of word recognition accuracies with GMMs of different sizes and fixed number of tied states are given in Table 1.3 Subsequently the number of tied states is also varied for 16 and 32 GMMs/state systems as given in Table 1.4. As the SDS system with 16 and 32 GMMs/state are found to give similar performances and hence we have used 16 GMMs/state in the deployed system.

Table 5.2: Word accuracies with different GMM sizes and fixed no. of tied states

Decoder	No. of GMM/state				Total Test files
	4	8	16	32	
Voice Command	66.73	68.45	62.69	62.63	1000

Table 5.3: Word accuracies with different no. of tied states and 16 GMM vs 32 GMM

Decoder	Sennon								Tot. Test files
	1000		1500		2000		2500		
	16 GMM	32 GMM	16 GMM	32 GMM	16 GMM	32 GMM	16 GMM	32 GMM	
Voice Command	70.23	69.5	71.34	62.8	72.38	72.8	72.69	72.6	1000

5.4 Speech Recognition with SPHINX

5.4.1 SPHINX Train

After preparing the environment and data preparation for speech recognition, we have prepared the sample data for training that already collected.

We have followed the following steps [56]:

- In the workspace directory we create the directory hmm using the command mkdir hmm.
- After that we have to go to the hmm directory using the command cd hmm then execute the following command

  $SPHINXTRAIN/scripts_pl/setup SphinxTrain.pl -task calflow1

  The above command sets up all the folders and files required for training. The above script generates the following important directories in hmm directory which contains the configuration files and the required transcript and dictionary files.
- After executing above command, we go to the hmm dir using the command cd hmm
- Then executing the following command

  perl scripts_pl/make_feats.pl -ctletc/calflow1_train.fileids

  The above command extracts the feature in the form of .mfc files and saves them in the feat directory (/workspace/hmm/etc).

- Then we run the command

  $ perl scripts_pl/RunAll.pl

  The whole process may take some time. This creates the trained vectors which will be used while decoding.

5.4.2 SPHINX Decode

After completion training, we have prepared the sample data for decoding that already collected. We have followed the following steps:

- We make the decode directory in the path /home/WordModel/workspace.
- In the path /home/WordModel/workspace/decode

  Then we create the subfolders by name feats, models and wav folders where,

a) feats directory contains all the .mfc (feature extracted files) files

b) wav directory contains all the test wav files

- In the path /home/WordModel/workspace/decode/models create the subfolders hmm and lm

  a) hmm directory has the following files created during training: feat.params, mdef, means, mixture_weights, transition_matrices, variances

  b) lm directory has the following files:

calflow1.dic, calflow1.filler and the calflow1.fsg files

All the above files in hmm and lm are copied from the trained models executing the following commands in the path /home/WordModel/workspace/

```
cp hmm/model_parameters/taskword.cd_cont_1000_8/* decode/models/hmm/
cp hmm/model_architecture/taskword .1000.mdef decode/models/hmm/mdef
cp hmm/etc/calflow1.dic decode/models/lm
cp hmm/etc/calflow1.filler decode/models/lm
cp hmm/etc/feat.paramsdecode/models/hmm
```

- We have to make calflow1.jsgf file in the lm directory which has the following format

```
#JSGF V1.0;
public<topping>
<words> = (on | off | status|fan|light|motor|freeze|ac|yes|no|)
```

- Now we run the following command to create the calflow1.sfg file

```
/home/WordModel/workspace/tools/sphinxbase/bin/sphinx_jsgf2fsg
calflow1.jsfg >calflow1.fsg
```

- The feat.params should have the following enteries–

```
 alpha 0.97
-samprate 8000
-frate 100
-dither yes
-doublebw no
```

```
-nfilt 31
-ncep 13
-lowerf 200
-upperf 3500
-nfft 512
-wlen 0.0256
-transform legacy
```

- We use following Feature Extraction Command for Testing

  /home/WordModel/workspace/tools/sphinxbase-0.6/bin/sphinx_fe-argfile models/hmm/feat.params -c test_files -di wav/ -do feats -ei wav -mswav yes –eomfc

  Where,

  -argfile is specified as models/hmm/feat.params. -mswav specifies the Microsoft Wave file format. test_files contains the list of all the wave files used for testing prepared is a manner similar to that of training.

- The command for decoding is as follows

  /home/WordModel/workspace/tools/sphinx3-0.8/bin/sphinx3_decode -hmm models/hmm -op_mode 2 -fsg models/lm/assamese2.fsg –dict models/lm/calflow1.dic -fdict models/lm/calflow1.filler -ctltest_files -logfn log.txt -hyp /dev/ttyUSB1 -cepdir feats/

Where,

-hmm Directory for specifying Sphinx 3's hmm, the following files are assummed to be present, mdef, mean, var, mixw, tmat. If -mdef, -mean, -var, -mixw or -tmat are specified, they will override this command. -op_mode Operation mode, for internal use only. Since FSG is the mode used so -op_mode has to be set to 2 -fsg Finite state grammar. -ctl Control file listing utterances to be processed (List of file that has to processed) -logfn Log file (log.txt) -hyp Recognition result file, with only words (out.txt) -cepdir Input cepstrum files directory (prefixed to file specs in control file) (Where, feats/ directory contain all the test mfc files.

5.4.3. Result Analysis

Matching Sound

```
INFO: utt.c(195): Processing: of121
INFO: feat.c(1148): At directory feats/
INFO: feat.c(378): Reading mfc file: 'feats//of121.mfc'[0..-1]
INFO: cmn.c(175): CMN:  4.85  4.07 -2.75  1.21 -0.52 -0.13  0.08 -0.77  0.19 -0.33
0.05 -0.06 -0.19
..............
INFO: fsg_search.c(1080): Utt of121: 134 frames, 679 HMMs evaluated, 1084
history entries

Backtrace(of121)
FV:of121>          WORD  SFrm  EFrm AScr(UnNorm)   LMScore  AScr+LScr
AScale
fv:of121>        <sil>    0    14      28918    -72912    -43994    146759
fv:of121>          off   15    48     260615    -34779    225836    457815
fv:of121>        <sil>   49   133    2995539    -72912   2922627   3338085
FV:of121>        TOTAL              3285072   -180603

FWDVIT: off (of121)
FWDXCT: of121 S 3922182 T 3104469 A 3285072 L -180603 0 28918 -72912 <sil>
15 260615 -34779 off 49 2995539 -72912 <sil> 134

INFO: stat.c(174):  134 frm;    4 cdsen/fr,   12 cisen/fr,    33 cdgau/fr,    96 cigau/fr,
Sen 0.00, CPU 0.00 Clk [Ovrhd 0.00 CPU 0.00 Clk];  Search: 0.00 CPU 0.00 Clk
(of121)
INFO: corpus.c(661): of121:    0.0 sec CPU,    0.0 sec Clk;  TOT:      0.0 sec CPU,
0.0 sec Clk
```

Non matching sounds:

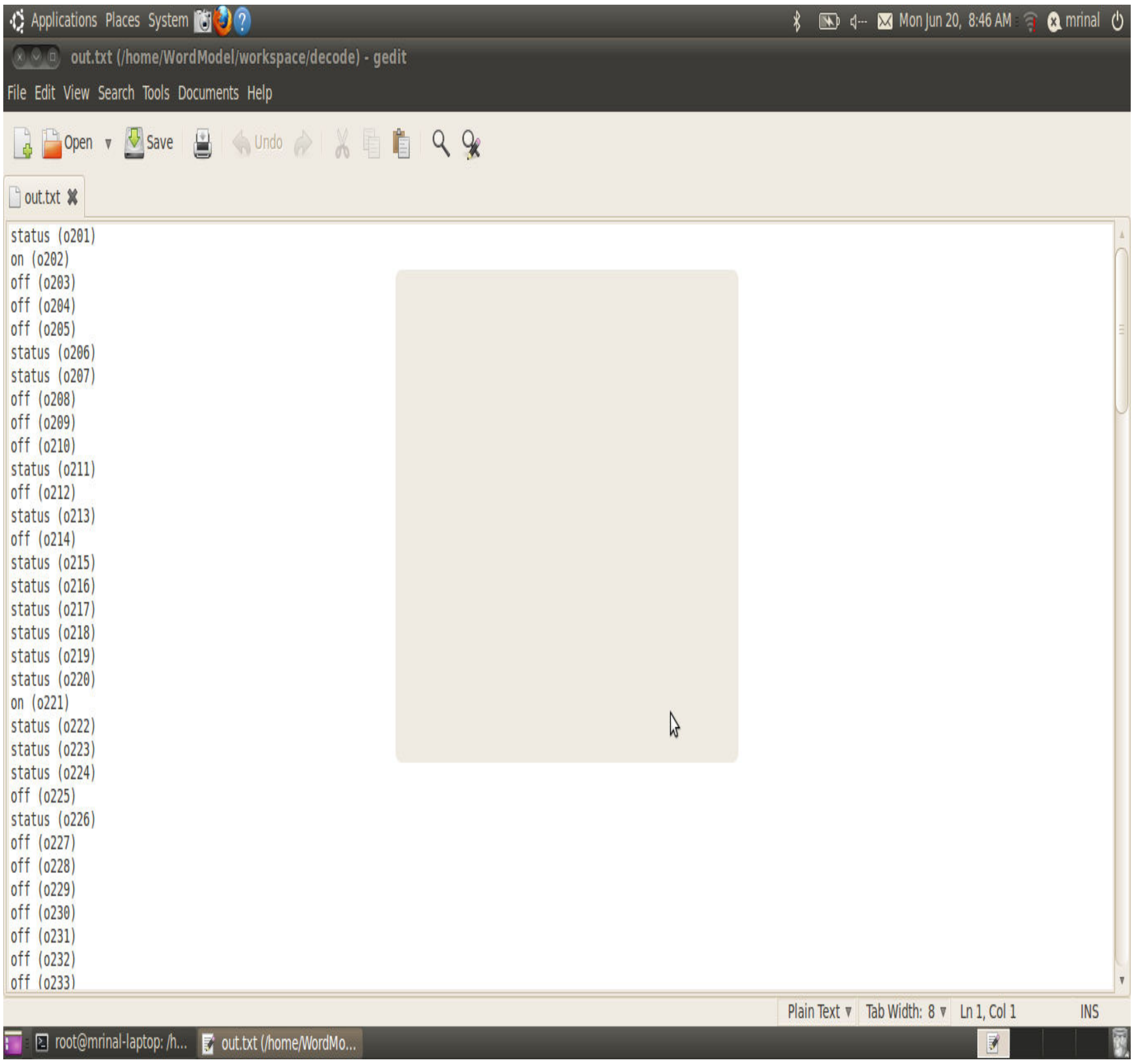
```
INFO: utt.c(195): Processing: o201
INFO: feat.c(1148): At directory feats/
```

```
INFO: feat.c(378): Reading mfc file: 'feats//o201.mfc'[0..-1]
INFO: cmn.c(175): CMN:  8.89  0.63 -0.69  0.14 -0.26 -0.29 -0.29 -0.06 -0.17 -0.02 -
0.16 -0.12 -0.08
.............
INFO: fsg_search.c(1080): Utt o201: 121 frames, 840 HMMs evaluated, 1275 history
entries

Backtrace(o201)
FV:o201>  WORD  SFrm  EFrm AScr(UnNorm)   LMScore  AScr+LScr    AScale
fv:o201>          <sil>    0   30      710597    -72912    637685    800995
fv:o201>         status   31   65     -253102    -34779   -287881    107181
fv:o201>          <sil>   66  120     2147051    -72912   2074139   2357378
FV:o201>           TOTAL               2604546   -180603

FWDVIT: status (o201)
FWDXCT: o201 S 3297574 T 2423943 A 2604546 L -180603 0 710597 -72912 <sil>
31 -253102 -34779 status 66 2147051 -72912 <sil> 121

INFO: stat.c(174):  121 frm;    6 cdsen/fr,   12 cisen/fr,    44 cdgau/fr,    96 cigau/fr,
Sen 0.00, CPU 0.00 Clk [Ovrhd 0.00 CPU 0.00 Clk];  Search: 0.00 CPU 0.00 Clk
(o201)
INFO: corpus.c(661): o201:     0.0 sec CPU,     0.0 sec Clk;  TOT:       0.0 sec CPU,
0.0 sec Clk
```

Silence sounds

```
INFO: utt.c(195): Processing: o111
INFO: feat.c(1148): At directory feats/
INFO: feat.c(378): Reading mfc file: 'feats//270320.mfc'[0..-1]
INFO: cmn.c(175): CMN:  6.39  0.30 -0.02 -0.29 -0.13 -0.25 -0.24 -0.07 -0.24  0.10 -
0.01 -0.02  0.06 .......
INFO: fsg_search.c(1080): Utt 270320: 70 frames, 1781 HMMs evaluated, 348 history
entries
WARNING: "fsg_search.c", line 949: No history entry in the final frame 69; using last
```

```
entry at frame 57ERROR: "fsg_search.c", line 1001: Final state not reached;
backtracing from best scoring entry
Backtrace(270320)
FV:270320>              WORD  SFrm  EFrm AScr(UnNorm)    LMScore  AScr+LScr
AScale
fv:270320>             <sil>    0    20    -1096388    -72912   -1169300    -882301
fv:270320>             <sil>   21    57     -208573    -72912    -281485    379607
FV:270320>              TOTAL              -1304961    -145824
FWDVIT: (0111)
FWDXCT: o111 S -191116 T -1450785 A -1304961 L -145824 0 -1096388 -72912
<sil> 21 -208573 -72912 <sil> 70
INFO: stat.c(174):   70 frm;   100 cdsen/fr,  130 cisen/fr,   800 cdgau/fr,  1040 cigau/fr,
Sen 0.01, CPU 0.02 Clk [Ovrhd 0.01 CPU 0.01 Clk];  Search: -0.00 CPU 0.00 Clk
(270320)
INFO: corpus.c(661): 270320:    0.0 sec CPU,    0.0 sec Clk;  TOT:       2.2 sec CPU,
2.2 sec Clk.
```



Applications Places System
Mon Jun 20, 8:46 AM
mrinal
out.txt (/home/WordModel/workspace/decode) - gedit
File Edit View Search Tools Documents Help
Open
Save
Undo
out.txt
Plain Text
Tab Width: 8
Ln 1, Col 1
INS
root@mrinal-laptop: /h...
out.txt (/home/WordMo...

```
status (o201)
on (o202)
off (o203)
off (o204)
off (o205)
status (o206)
status (o207)
off (o208)
off (o209)
off (o210)
status (o211)
off (o212)
status (o213)
off (o214)
status (o215)
status (o216)
status (o217)
status (o218)
status (o219)
status (o220)
on (o221)
status (o222)
status (o223)
status (o224)
off (o225)
status (o226)
off (o227)
off (o228)
off (o229)
off (o230)
off (o231)
off (o232)
off (o233)
```

Figure 5.6: Screen shot of output result

Result displayed in manually:

If one wav file is testing then the result will be displayed as-

word recognition(wave file name)

example: on (o221)

Accuracy

From the result of the recognition phase, we have found average 67% accuracy, i.e. our testing voice data has matched with training voice data with overall 67%. This result indicates that the training of our system is completed successfully and shows that the developed system is speaker independent.

5.5 Outcome for Wi-Fi Communication

In this one of the important parts of present work has implemented internet of things with the help of Wi-Fi and established a few points.

Remote Controlling of Load

A device is built up that could control the power supply of the load connected to it, thereby allowing the user to remotely turn the device ON or OFF connected to the load using a web based application through Wi-Fi.

Parameter Based Automation

The present system has given a focus on automation of the power control of the load depending upon certain conditions being met by the parameter which we were collecting from the sensor.

Modular Sensor Based Automation

A system is provided for creating with a sense of generality and thereby decides to include a mechanism to automate the control of the load by introducing some sensors which would be modular in nature. Thus, we could connect any load and automate it on the basis of any parameters just by introducing a suitable sensor to the system.

5.5.1 Wi-Fi enabled Circuit Stimulation

The programming code for this IoT communication is tested first on the circuit stimulator software called Proteus ISIS before actually burning our code into the microcontroller which have helped us save a helped a lot in debugging and testing of code, as a result of which our final working code was possible to be executed successfully and getting the desired result from the system.

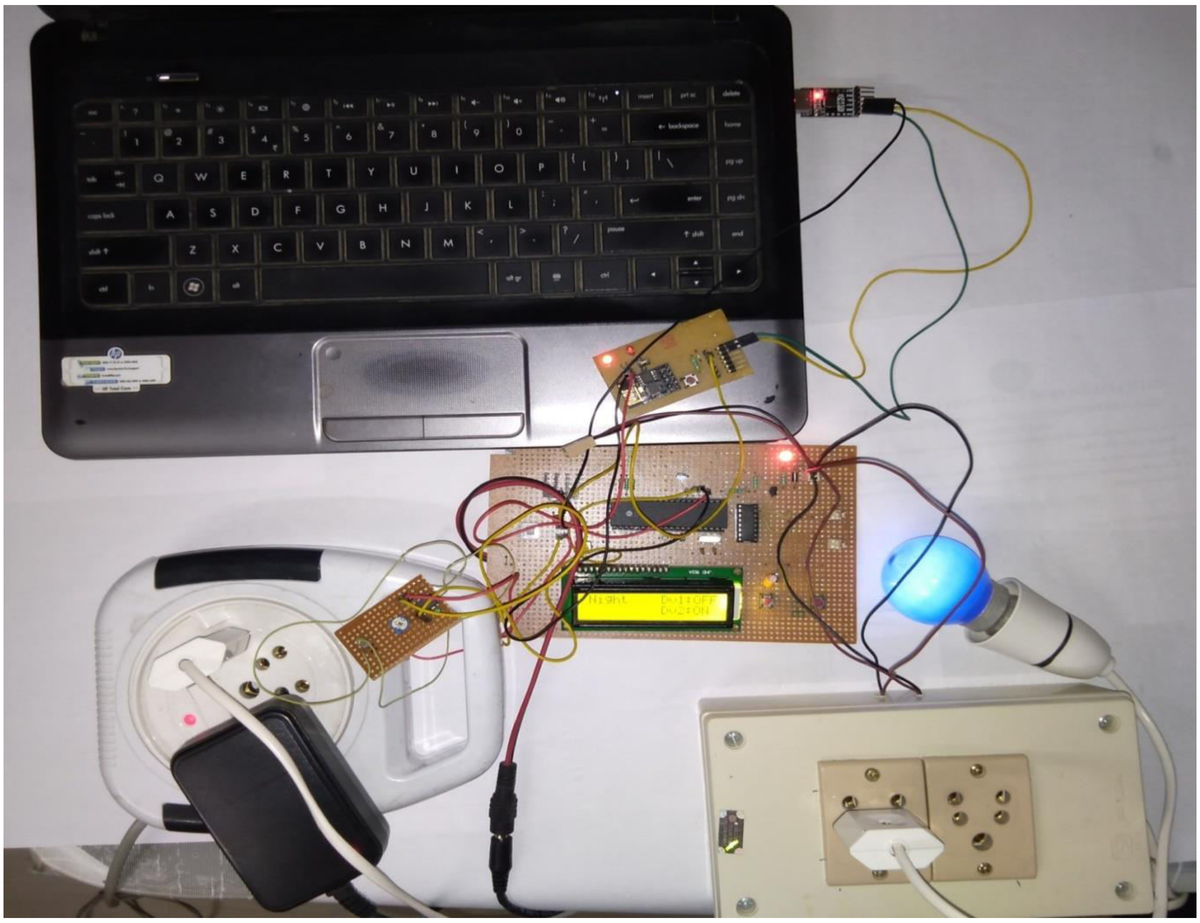
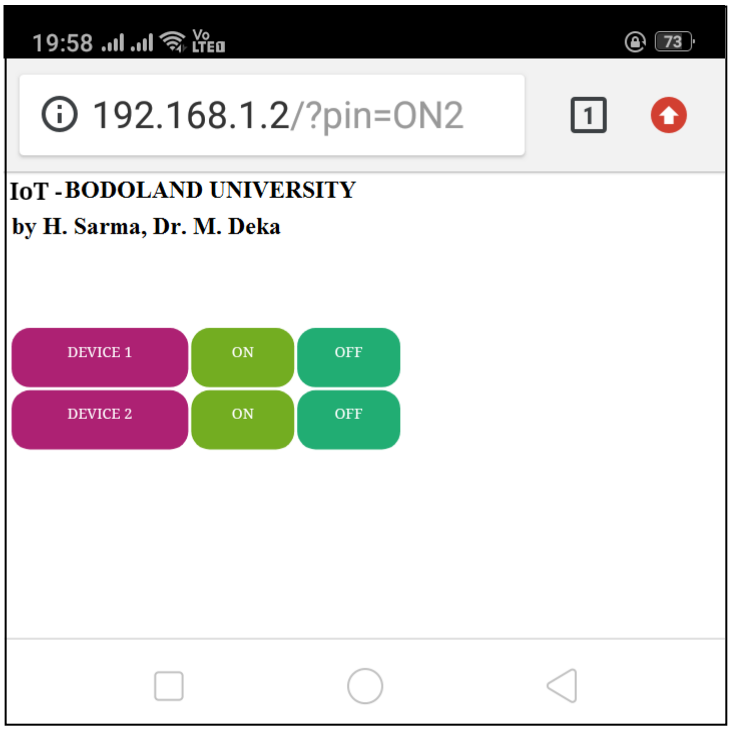


Figure 5.7: Screenshot for the IoT Communication in the server side



IoT -BODOLAND UNIVERSITY
by H. Sarma, Dr. M. Deka

Figure 5.8: Screenshot for Wi-Fi communication at User Side

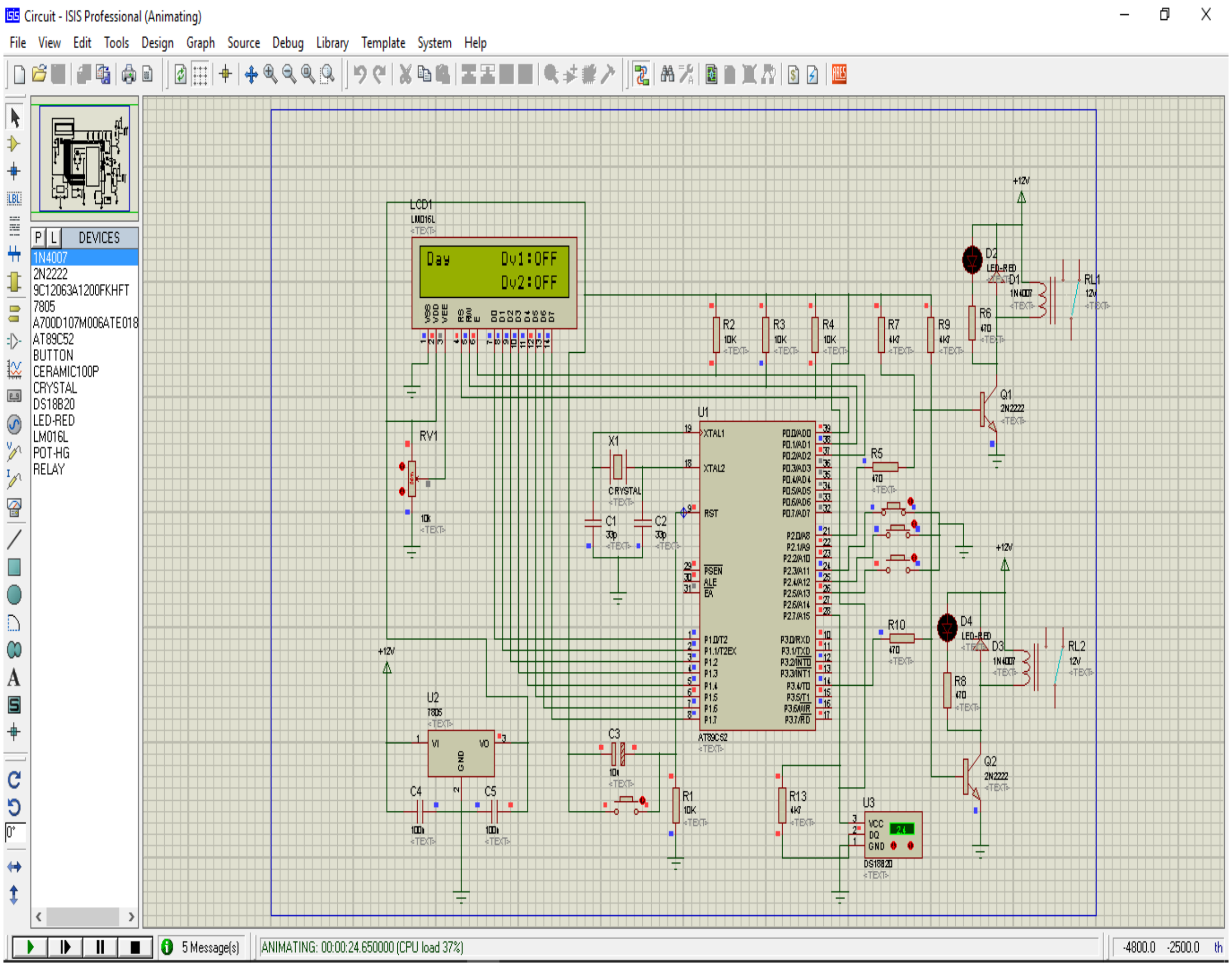


Figure 5.9: Screenshot of circuit simulation in Proteus ISIS