ABSTRACT

ABSTRACT

Internet of things is a technology of the future that has already started to touch our homes. Here we propose an IOT based home automation system using raspberry pi that automates home appliances and allows user to control them easily through internet from anywhere over the world. Our proposed system consists of a micro-controller-based circuit that has lights and fan connected to it along with LCD display and Wi-fi connector interfaced with raspberry pi. Our system interacts with out online IOT system that IOT Gecko free web interface for controlling our home appliances with ease. After linking with IOT Gecko, the user is allowed to send load switching commands over IOT to our circuit. The circuit receives the commands over IOT by connecting to internet using Wi-Fi connector and then the raspberry processor processes these commands. After this the processor now processes these instructions to get user commands. It then displays these on an LCD display. It operates the loads (lights and fan) for switch them on/off according to desired user commands. Thus we automate home appliances over internet using raspberry pi.

CHAPTER 1
INTRODUCTION

1.1 INTRODUCTION

The Internet of Things (IoT) is the network of physical objects or "things" embedded with electronics, software, sensors, and network connectivity, which enables these objects to collect and exchange data . In home automation smart devices and sensors that sense the physical experience and convert into stream of information data. The major element of home automation based on IoT is sensor network and raspberry pi. Sensor networks are used for sensing and monitoring while raspberry pi collect the data monitor the data and depends on collected mange the device like fan, light, door motion and opening-closing of curtains. Suppose the ambient light is less that I am going to feel darkness then according to ambient light its automatically open the curtains

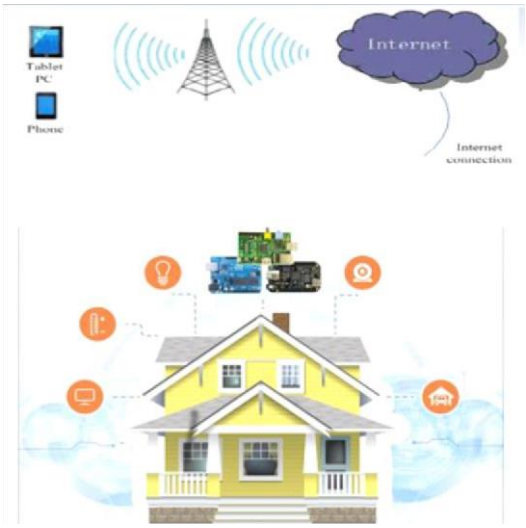


Fig. 1. Complete home automation using Raspberry pi

Thus, home automation can be defined as a mechanism removing as much human interaction as technically possible and desirable in various domestic processes and replacing them with programmed electronic systems. Ultimately it is a system that aims to heighten quality of life with the automation of household activity that may be controlled over the Internet or telephone.

1.2 RASPBERRY PI

The Raspberry Pi is a series of credit card–sized singleboard computers developed in the United Kingdom by the Raspberry Pi Foundation with the intention of promoting the teaching of basic computer science.

The Raspberry Pi has four distinct power modes:

- The run mode:
  - The central processing unit (CPU) and all functionality of the ARM11 core are available and powered up.

- The standby mode:
  - The main core clocks are shut down (the parts of the CPU that process instructions are no longer running) although the power circuits on the core are still active. In this mode, known as ‘‘Wait for Interrupt’’ (WFI) mode, the core can be quickly woken up by a process generating a special call to the CPU called an interrupt. This interrupt will stop any current processing and do what the calling process has asked for.
- The shutdown mode:
  - There is no power.
- The dormant mode:
  - The core is powered down and all caches are left powered on.

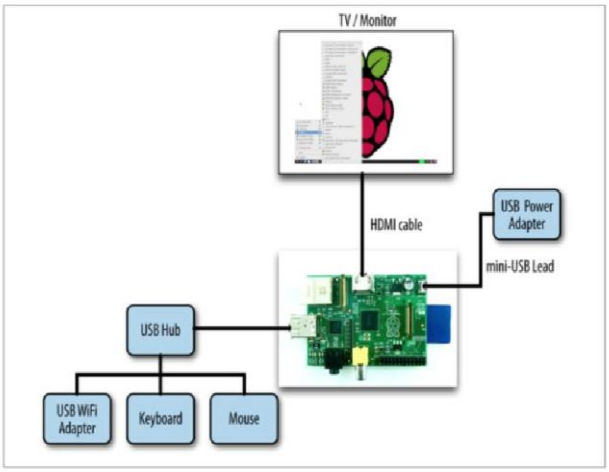


Fig. 1.2. Typical Raspberry pi system

Raspberry Pi is an open source hardware technology combined with a programming language and an Integrated Development Environment (IDE).

The Raspberry Pi platform allows the user to create custom hardware and applications to control it via its namesake programming language.

Many software languages are available on the Raspberry Pi and I am interested in four. These are the C++, Python, SQL, and HTSQL. C++ uses for programming Arduino. HTSQL (Hyper Text Structured Query Language) to provide a web interface to database that is easy to query via the web browser. Operating systems that are available to install the Raspberry is Raspbian. Raspbian is based upon the Debian Wheezy Linux operating system and has been optimized for use with Raspberry Pi. The Raspberry Pi is connected to the Raspberry Pi's GPIO pins, and with the inclusion of the software, it will be able to communicate between our electronic devices, the Raspberry Pi's operating system, and web-based propose model.

CHAPTER 2
SYSTEM ANALYSIS

2.1 EXISTING SYSTEM DISADVANTAGES

- It uses Arduino UNO board basic model which is cheaper, the load operation is limited to 6.
- Only a single program can run at a given time.
- In very archaic buildings rewiring of loads and making it centralized near to Master PC may be a bit hassle.
- Not compatible with IOS devices.
- Has signal limitation.

2.2 PROPOSED SYSTEM

We have used Raspberry pi as the main base of our project for processing data. Our aim is to build a fully workable and controllable home appliance on circuit switch board directly. The user interface would be created for desktops or laptops.

The appliances to be operated are connected to the GPIO pins of the raspberry pi, through a breadboard and switches.

2.3 ADVANTAGES OF PROPOSED SYSTEM

- This system over comes the signal limitation of the existing system.
- This system uses raspberry pi to achieve automation unlike existing system.
- This system can connect the user interface to the raspberry pi system in such a way that it we can switch on the devices connected from anywhere.
- Energy efficient.
- No cooling systems needed.
- Allows multiple appliances/programs to run at once.
- UI is stored in the cloud.
- Easy to use.

CHAPTER 3
FEASBILITY REPORT

3.1 UNDERSTANDING FEASIBILITY:

Feasibility study means the analysis of problem to determine if it can be solved effectively. In other words it is the study of the possibilities of the proposed system it studies the work ability, impact on the organization ability to meet user's need and efficient use of resources. Three aspects in which the system has to be feasible are:-

3.1.1 ECONOMICAL FEASIBILITY:

The economic analysis checks for the high investment incurred on the system. It evaluates development & Implementing charges for the proposed "Home Automation Using Raspberry Pi". The Software used for the development is easily available at minimal cost & the database applied is freely available hence it results in low cost implementation**.**

3.1.2 TECHNICAL FEASIBILITY:

This aspect concentrates on the concept of using Computer Meaning, "Mechanization" of human works. Thus the automated solution leads to the need for a technical feasibility study. The focus on the platform used database management and users for that Software. The proposed system doesn't require an in depth technical knowledge as the system development is simple and easy to understand. The Software used makes the system user friendly (GUI).

3.1.3 BEHAVIOURAL FEASIBILITY:

Behavioural feasibility deals with the runtime performance of the Software, the proposed system must score higher than the present in the behavioural study. The Software should have end user in mind when the system is designed while designing Software the programmer should be aware of the conditions user's Knowledge input, output, calculations etc

CHAPTER 4
REQUIREMENTS SPECIFICATION

4.1 HARDWARE REQUIREMENTS

Client Side	Server Side
Raspberry pi 2 or above	Processor: Intel i3 and above
LED Lights	R.A.M: 2GB and above
Fan	Ethernet Card
Bread Board	
Jumper Cables	
Wi-Fi adapter	
Relay	

4.2 SOFTWARE REQUIREMENTS

Client Side	Server Side
Raspbian OS	Xampp
Python Terminal or IDE	Notepad++
	Web Browser

4.3 TECHNOLOGIES USED

Client Side	Server Side
Python 3 or above	PHP
urllib3 package	HTML
rpi.GPIO package	Ajax
	JavaScript
	CSS
	Bootstrap

4.4 PHP

- PHP is an acronym for "PHP: Hypertext Preprocessor"
- PHP is a widely-used, open source scripting language
- PHP scripts are executed on the server
- PHP is free to download and use

PHP is an amazing and popular language!

It is powerful enough to be at the core of the biggest blogging system on the web (WordPress)
It is deep enough to run the largest social network (Facebook)
It is also easy enough to be a beginner's first server side language

4.4.1 WHAT IS A PHP FILE?

- PHP files can contain text, HTML, CSS, JavaScript, and PHP code.

- PHP code are executed on the server, and the result is returned to the browser as plain HTML.
- PHP files have extension ".php".

4.4.2 WHAT CAN PHP DO?

- PHP can generate dynamic page content
- PHP can create, open, read, write, delete, and close files on the server
- PHP can collect form data
- PHP can send and receive cookies
- PHP can add, delete, modify data in your database
- PHP can be used to control user-access
- PHP can encrypt data

With PHP you are not limited to output HTML. You can output images, PDF files, and even Flash movies. You can also output any text, such as XHTML and XML.

4.4.3 WHY PHP?

- PHP runs on various platforms (Windows, Linux, Unix, Mac OS X, etc.)
- PHP is compatible with almost all servers used today (Apache, IIS, etc.)
- PHP supports a wide range of databases.
- PHP is free. Download it from the official PHP resource: www.php.net
- PHP is easy to learn and runs efficiently on the server side.

4.4.4 EXAMPLE:

```
<!DOCTYPE HTML>
<html>
    <head>
        <title>Example</title>
    </head>
    <body>

        <?php
            echo "Hi, I'm a PHP script!";
        ?>

    </body>
</html>
```

4.5 HTML

Web pages are written in HTML - a simple scripting language. HTML is short for Hyper Text Markup Language**.**

• Hypertext is simply a piece of text that works as a link.

• Markup Language is a way of writing layout information within documents.

Basically an HTML document is a plain text file that contains text and nothing else. When a browser opens an HTML file, the browser will look for HTML codes in the text and use them to change the layout, insert images, or create links to other pages. Since HTML documents are just text files they can be written in even the simplest text editor. A more popular choice is to use a special HTML editor - maybe even one that puts focus on the visual result rather than the codes - a so-called WYSIWYG editor ("What You See Is What You Get").

Some of the most popular HTML editors, such as FrontPage or Dreamweaver will let you create pages more or less as you write documents in Word or whatever text editor you're using. However, there are some very good reasons to create your own pages - or parts of them – by hand. It is possible to create Web pages without knowing anything about the HTML source behind the page. There are excellent editors on the market that will take care of the HTML parts.

All you need to do is layout the page.

4.5.1 HTML TAGS:

- HTML tags are used to mark-up HTML elements.
- HTML tags are surrounded by the two characters < and > are called angle brackets, the first tag in a pair is the start tag and the second tag is the end tag.
- The text between the start and end tags is the element content.
- HTML tags are not case sensitive; <b> means the same as <B>.

4.6 AJAX

AJAX is a developer's dream, because you can:

- Update a web page without reloading the page
- Request data from a server - after the page has loaded
- Receive data from a server - after the page has loaded
- Send data to a server - in the background

4.6.1 WHAT IS AJAX?

AJAX = **A**synchronous **J**avaScript **A**nd **X**ML.
AJAX is not a programming language.
AJAX just uses a combination of:

- A browser built-in XMLHttpRequest object (to request data from a web server)
- JavaScript and HTML DOM (to display or use the data)

AJAX is a misleading name. AJAX applications might use XML to transport data, but it is equally common to transport data as plain text or JSON text.

AJAX allows web pages to be updated asynchronously by exchanging data with a web server behind the scenes. This means that it is possible to update parts of a web page, without reloading the whole page.

4.6.2 HOW AJAX WORKS

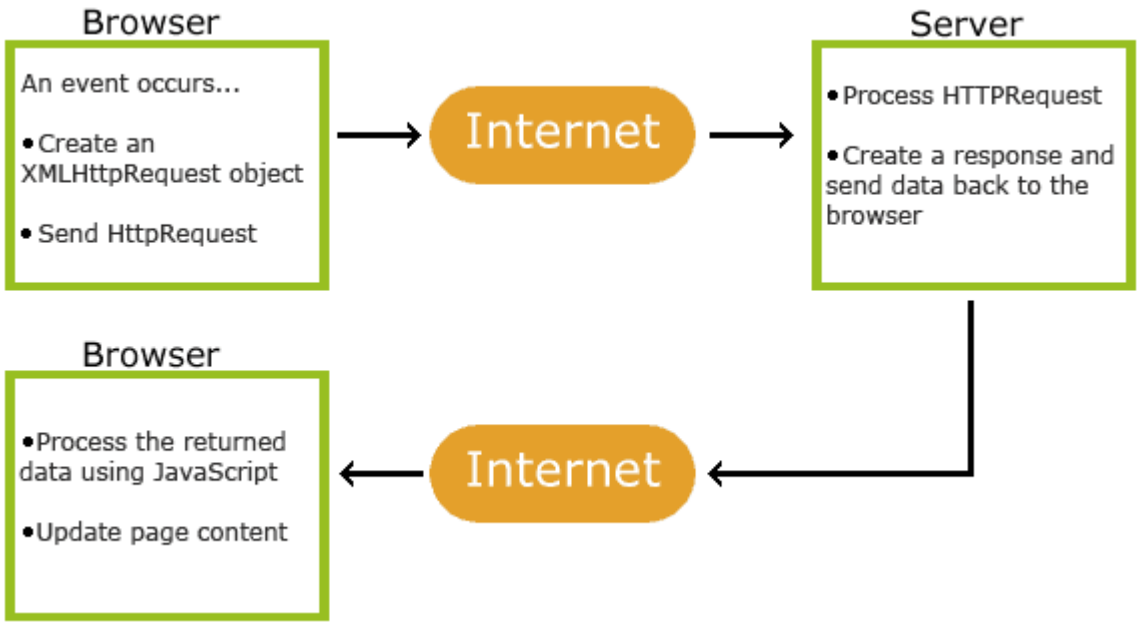


Fig. 4.1. Ajax Working

- 1. An event occurs in a web page (the page is loaded, a button is clicked)
- 2. An XMLHttpRequest object is created by JavaScript
- 3. The XMLHttpRequest object sends a request to a web server
- 4. The server processes the request
- 5. The server sends a response back to the web page
- 6. The response is read by JavaScript
- 7. Proper action (like page update) is performed by JavaScript

4.7 JAVASCRIPT

JavaScript is a scripting language that will allow you to add real programming to your Web pages. You can create small application type processes with JavaScript, like a calculator or a primitive game of some sort. However, there are more serious uses for JavaScript:

• **Browser Detection**:

Detecting the browser used by a visitor at your page. Depending on the browser, another page specifically designed for that browser can then be loaded.

• **Cookies**:

Storing information on the visitor's computer, then retrieving this information automatically next time the user visits your page. This technique is called "cookies".

• Control Browsers:

Opening pages in customized windows, where you specify if the browser's buttons, menu line, status line or whatever should be present.

• Validating Forms:

Validating inputs to fields before submitting a form. An example would be validating the entered email address to see if it has a @ in it, since if not, it's not a valid address.

4.8 PYTHON

Python is an easy to learn, powerful programming language. It has efficient high-level data structures and a simple but effective approach to object-oriented programming. Python's elegant syntax and dynamic typing, together with its interpreted nature, make it an ideal language for scripting and rapid application development in many areas on most platforms.

The Python interpreter and the extensive standard library are freely available in source or binary form for all major platforms from the Python Web site, https://www.python.org/, and may be freely distributed. The same site also contains distributions of and pointers to many free third party Python modules, programs and tools, and additional documentation.

The Python interpreter is easily extended with new functions and data types implemented in C or C++ (or other languages callable from C). Python is also suitable as an extension language for customizable applications.

4.8.1 PYTHON INTERPRETER

The Python interpreter is usually installed as /usr/local/bin/python3.6 on those machines where it is available; putting /usr/local/bin in your Unix shell's search path makes it possible to start it by typing the command:

python3.6

to the shell. Since the choice of the directory where the interpreter lives is an installation option, other places are possible; check with your local Python guru or system administrator. (E.g., /usr/local/python is a popular alternative location.)

On Windows machines, the Python installation is usually placed in C:\Python36, though you can change this when you're running the installer. To add this directory to your path, you can type the following command into the command prompt in a DOS box:

set path=%path%;C:\python36

Typing an end-of-file character (Control-D on Unix, Control-Z on Windows) at the primary prompt causes the interpreter to exit with a zero exit status. If that doesn't work, you can exit the interpreter by typing the following command: quit().

The interpreter's line-editing features include interactive editing, history substitution and code completion on systems that support readline. Perhaps the quickest check to see whether command line editing is supported is typing Control-P to the first Python prompt you get. If it beeps, you have command line editing. If nothing appears to happen, or if ^P is echoed, command line editing isn't available; you'll only be able to use backspace to remove characters from the current line.

4.9 BOOTSTRAP

Bootstrap is a powerful front-end framework for faster and easier web development. It includes HTML and CSS based design templates for common user interface components like Typography, Forms, Buttons, Tables, Navigations, Dropdowns, Alerts, Modals, Tabs, Accordion, Carousel and many other as well as optional JavaScript extensions.

Bootstrap also gives you ability to create responsive layout with much less efforts.

4.9.1 ADVANTAGES OF BOOTSTRAP

The biggest advantage of using Bootstrap is that it comes with free set of tools for creating flexible and responsive web layouts as well as common interface components.

Additionally, using the Bootstrap data APIs you can create advanced interface components like Scrollspy and Typeaheads without writing a single line of JavaScript.

Here are some more advantages, why one should opt for Bootstrap:

- **Save lots of time** — You can save lots of time and efforts using the Bootstrap predefined design templates and classes and concentrate on other development work.
- **Responsive features** — Using Bootstrap you can easily create responsive designs. Bootstrap responsive features make your web pages to appear more appropriately on different devices and screen resolutions without any change in markup.
- **Consistent design** — All Bootstrap components share the same design templates and styles through a central library, so that the designs and layouts of your web pages are consistent throughout your development.
- **Easy to use** — Bootstrap is very easy to use. Anybody with the basic working knowledge of HTML and CSS can start development with Bootstrap.
- **Compatible with browsers** — Bootstrap is created with modern browsers in mind and it is compatible with all modern browsers such as Mozilla Firefox, Google Chrome, Safari, Internet Explorer, and Opera.
- **Open Source** — And the best part is, it is completely free to download and use.

CHAPTER 5
SYSTEM DESIGN

5.1 INTRODUCTION

Systems design is the process or art of defining the architecture, components, modules, interfaces, and data for a system to satisfy specified requirements. One could see it as the application of systems theory to product development. There is some overlap and synergy with the disciplines of systems analysis, systems architecture and systems engineering

5.2 ARCHITECTURE

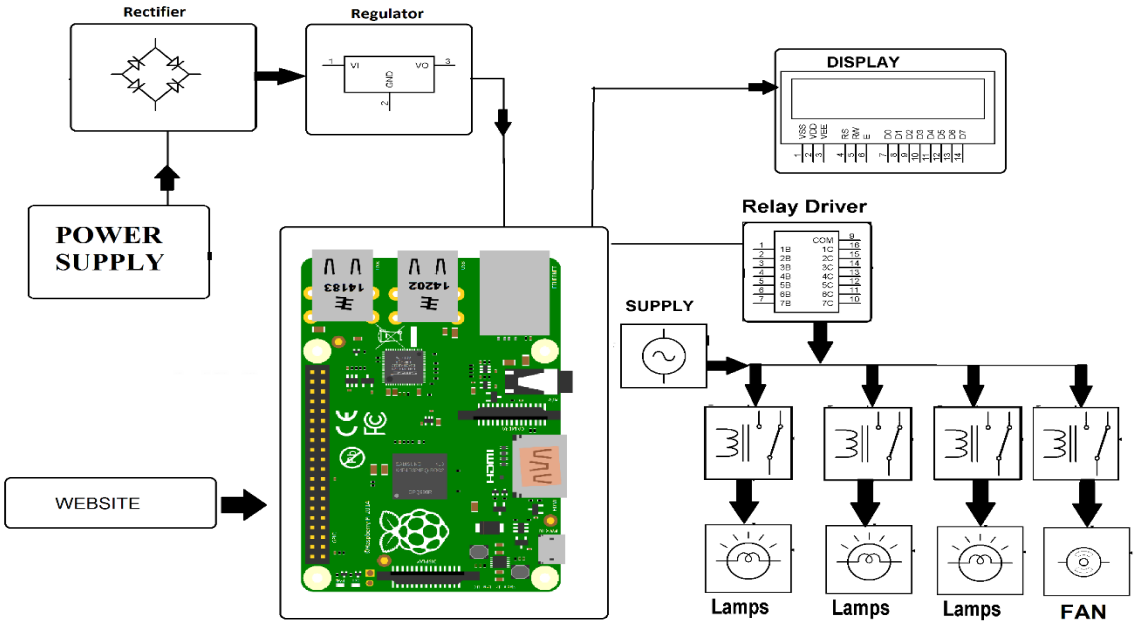


Fig. 5.1. System Architecture

5.3 MODULES

System consists of two modules:

1. Client 2. Server

5.3.1 CLIENT

- Raspberry Pi
- Electric Appliances

5.3.2 SERVER

- User Interface
- Button.php
- ButtonStatus.txt
- ButtonStatus.php

CHAPTER 6
UML DIAGRAMS

6.1 UNIFIED MODELLING LANGUAGE:

The Unified Modelling Language allows the software engineer to express an analysis model using the modelling notation that is governed by a set of syntactic, semantic and pragmatic rules. A UML system is represented using five different views that describe the system from distinctly different perspective. Each view is defined by a set of diagrams, which is as follows:

• **User Model View**:

i. This view represents the system from the user's perspective.
ii. The analysis representation describes a usage scenario from the end-users perspective.

• Structural model view:

i. In this model the data and functionality are arrived from inside the system.
ii. This model view models the static structures.

• **Behavioural Model View**:

i. It represents the dynamic behavioural as parts of the system, depicting the interactions of collection between various structural elements described in the user model and structural model view.

• **Implementation Model View**:

i. In this the structural and behavioural as parts of the system are represented as they are to be built.

• **Environmental Model View:**

i. In this the structural and behavioural aspects of the environment in which the system is to be implemented are represented.

6.2 UML DIAGRAMS

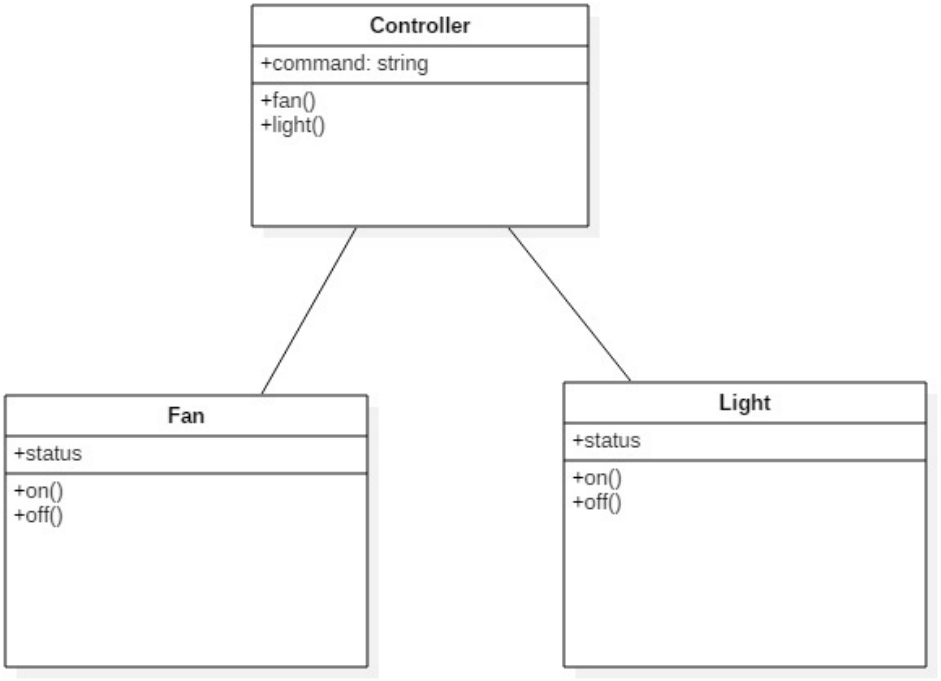


Fig. 6.2.1 Class Diagram

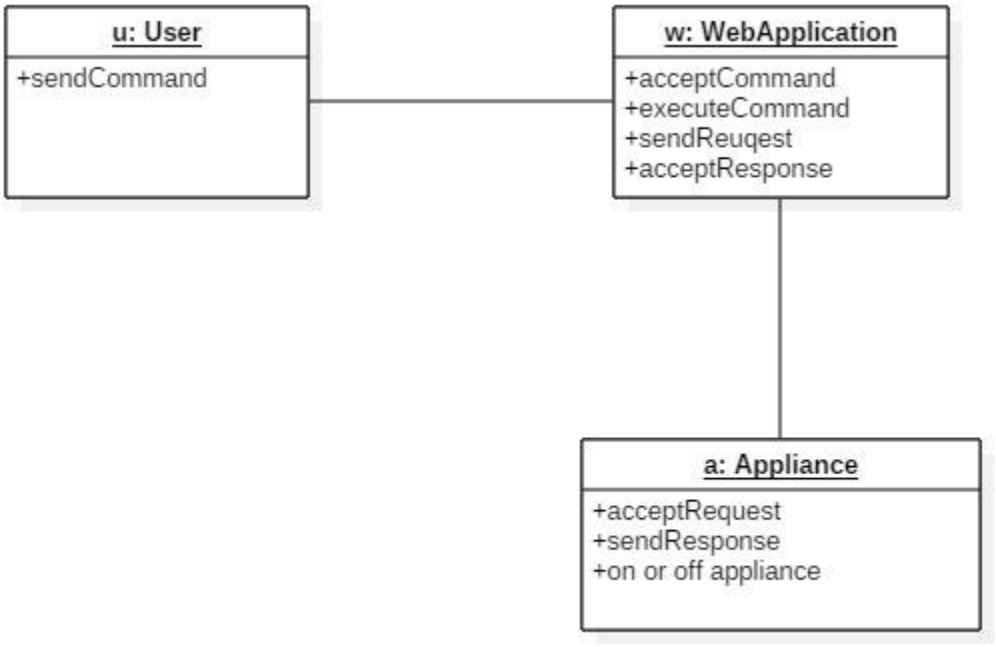


Fig. 6.2.2 Object Diagram

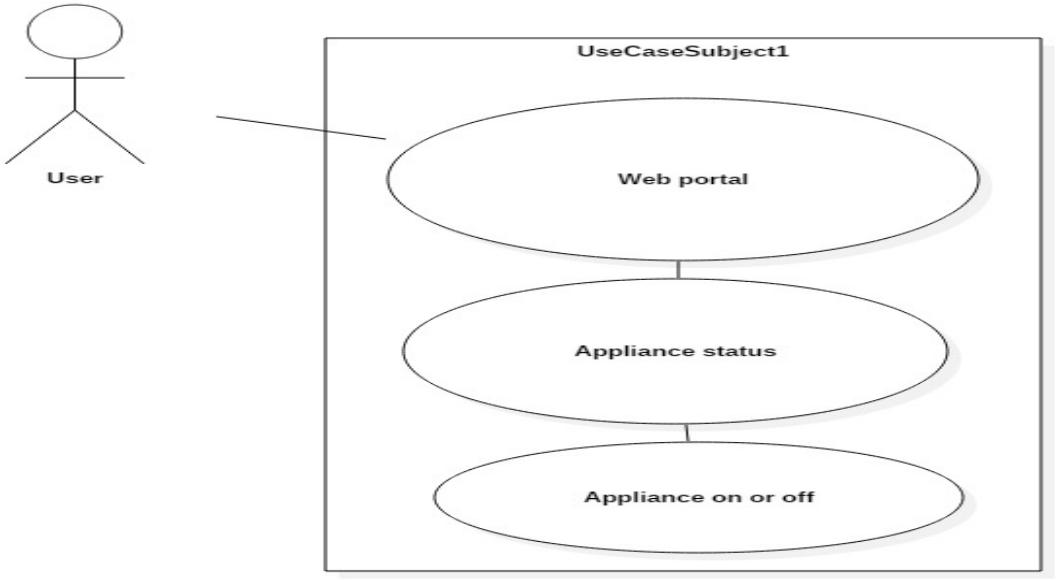


Fig. 6.2.3 Use Case Diagram

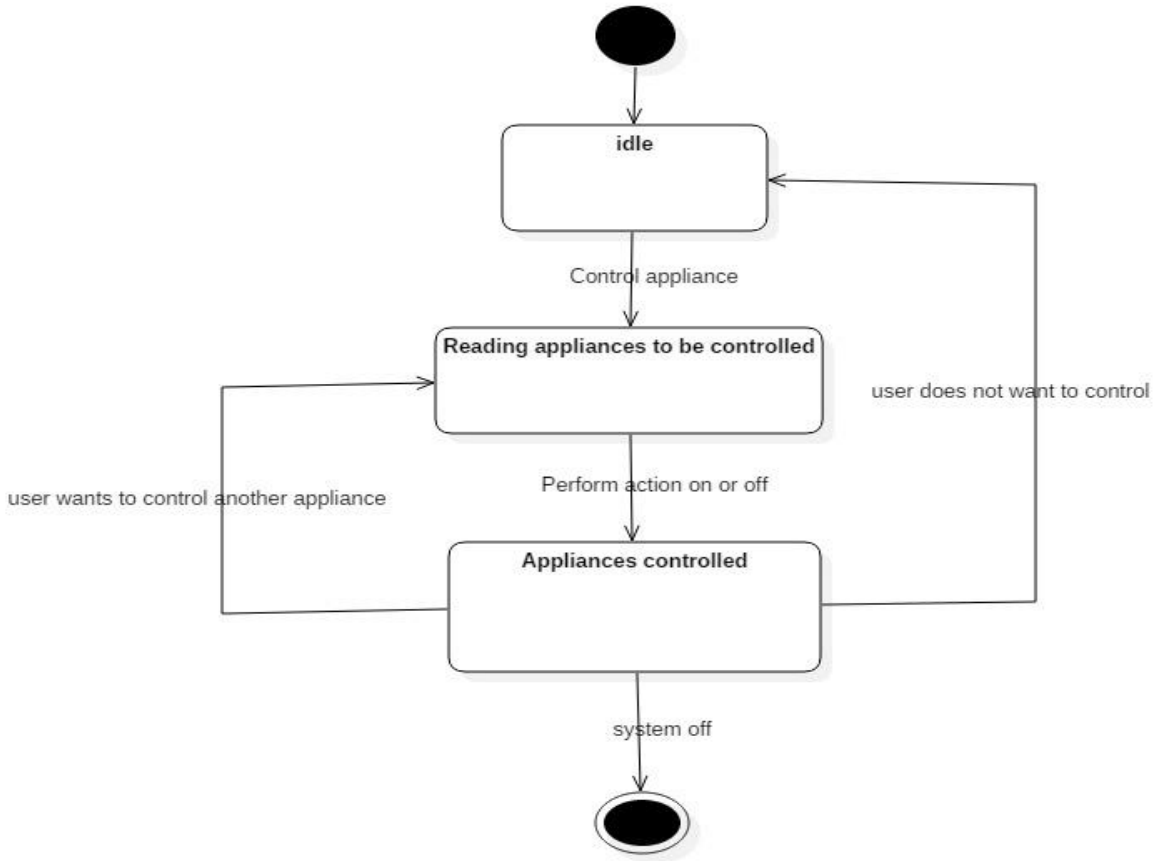


Fig. 6.2.4 State Chart Diagram

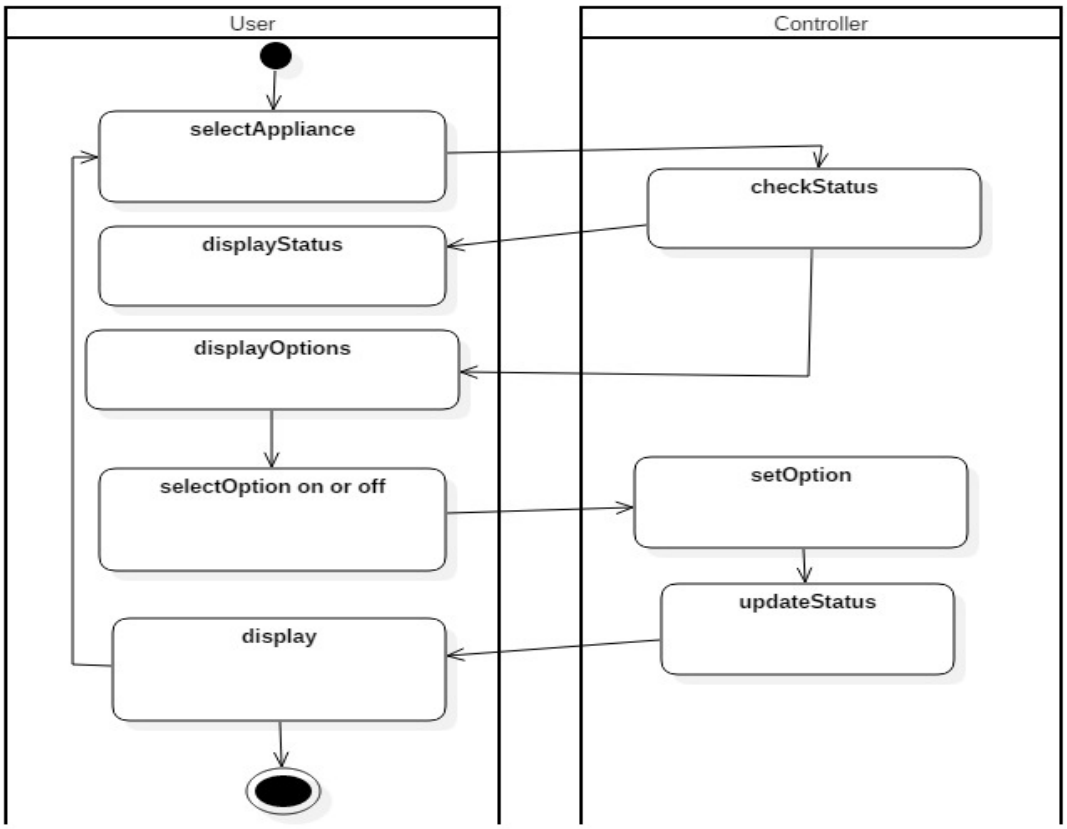


Fig. 6.2.5 Activity Diagram

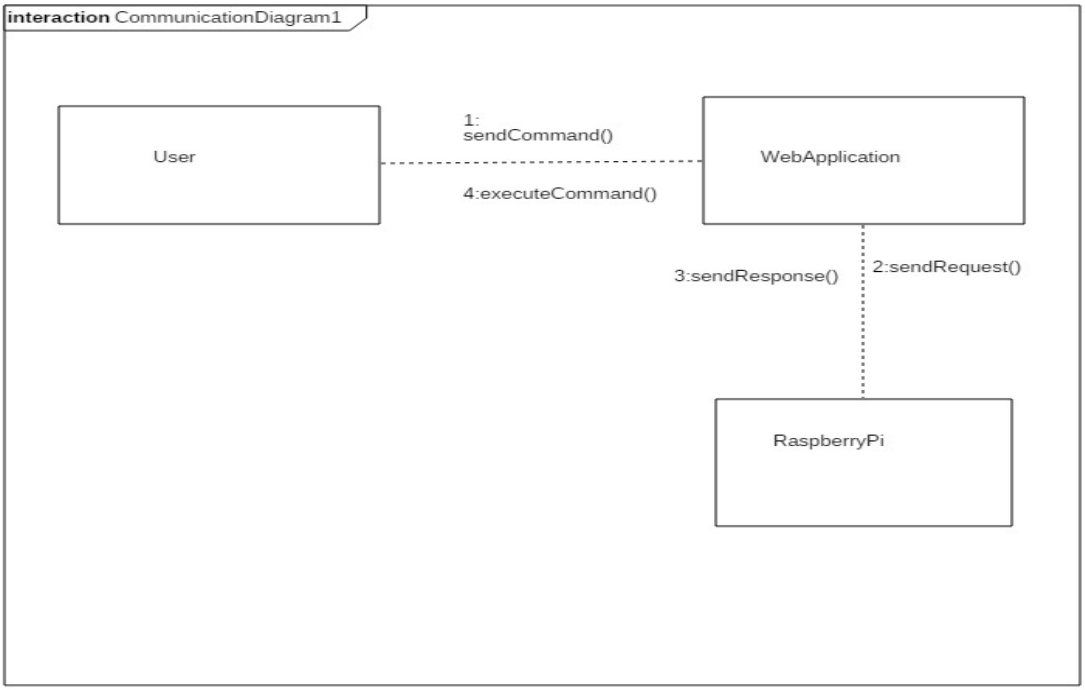


Fig. 6.2.6 Collaboration Diagram

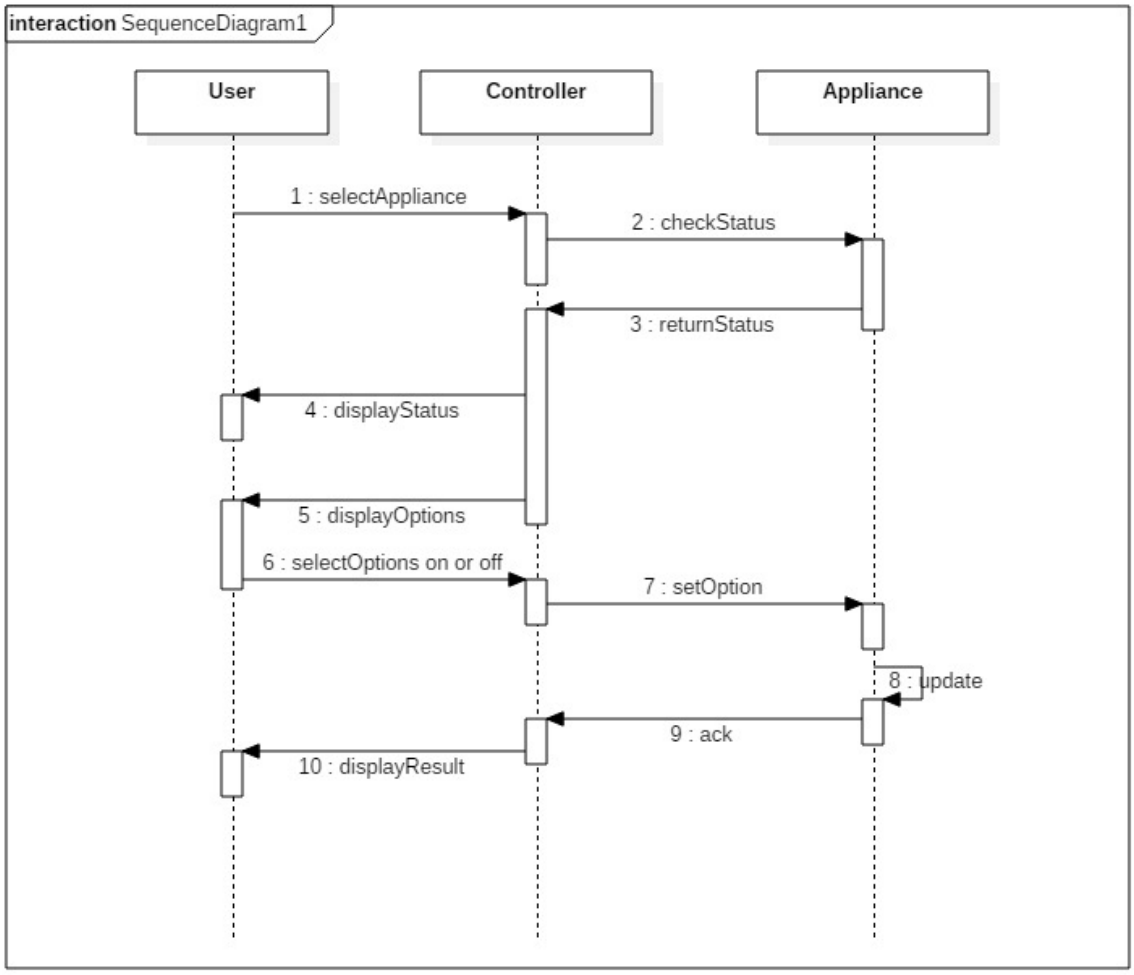


Fig. 6.2.7 Sequence Diagram

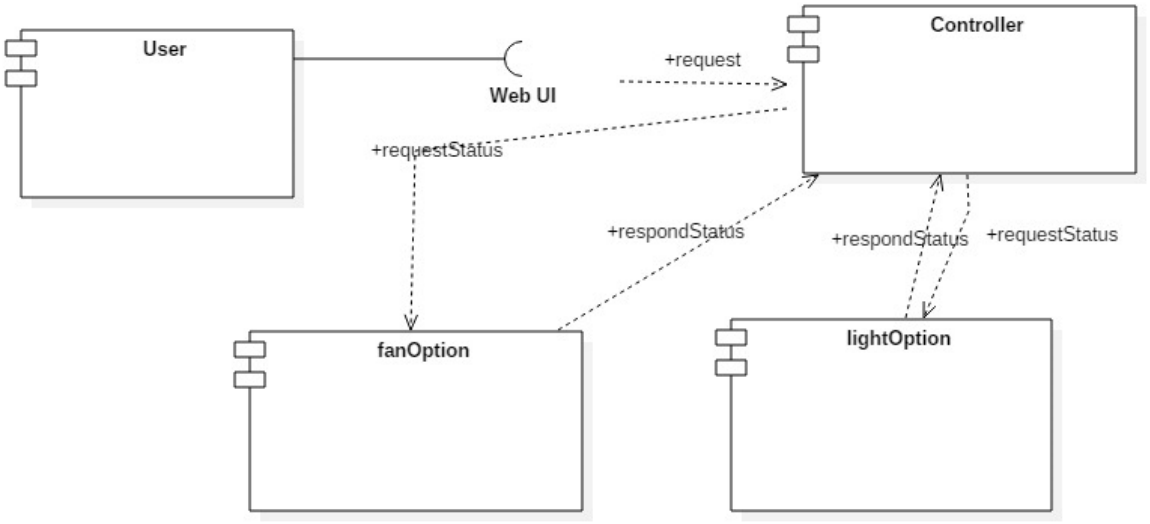


Fig. 6.2.8 Component Diagram

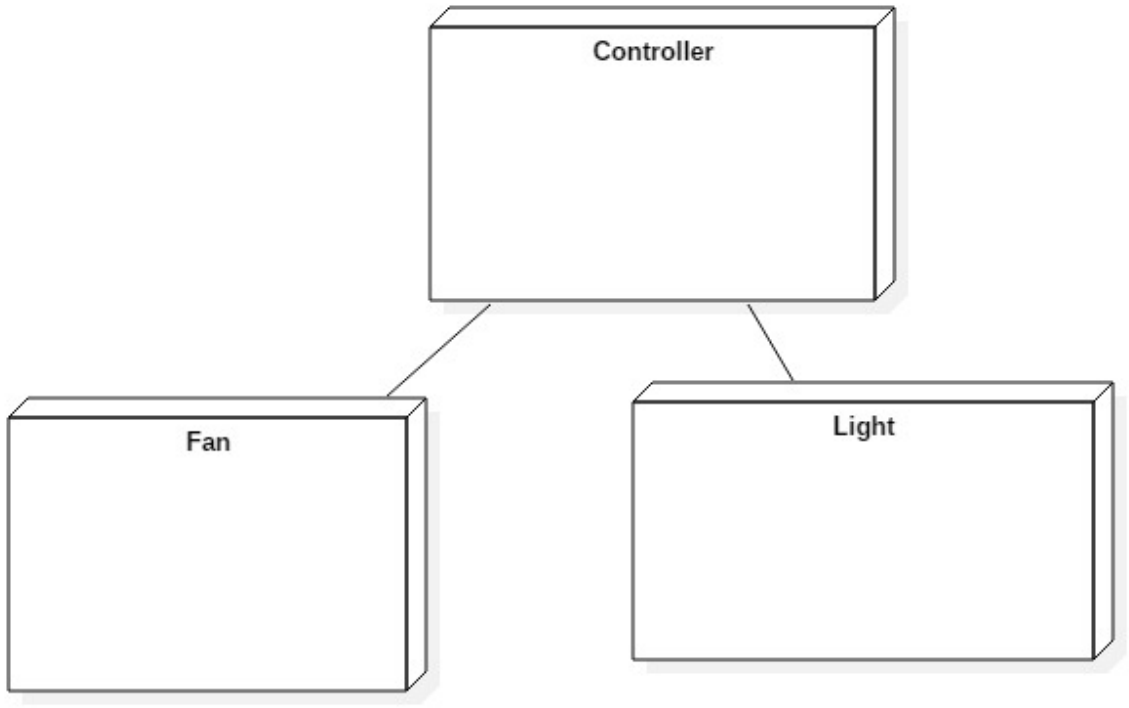


Fig. 6.2.9 Deployment Diagram

CHAPTER 7
CODING AND IMPLEMENTATION

7.1 INTRODUCTION:

Coding and Implementation is the stage of the project when the theoretical design is turned out into a working system. Thus it can be considered to be the most critical stage in achieving a successful new system and in giving the user, confidence that the new system will work and be effective**.**

7.2 WORKING / IMPLEMENTATION

You can consider the whole system to be composed of two parts:

- Server.
- Client.

The server is the web interface consisting of buttons and UI (User Interface) that will allow you to turn ON/OFF a device. It consists of PHP files, Html files and a .txt file (to store data).

The server usually stores information regarding the button press on the page (ON/OFF) on a .txt file. This is a simple Html file called main.html , consisting of two buttons.

The clicking of the buttons will trigger the execution of a PHP file called button.php.

This program serves as an API to store data on to a text file called buttonStatus.txt.

The data is a string :

- "ON",
  - if ON button is clicked
- "OFF"
  - if OFF button is clicked.

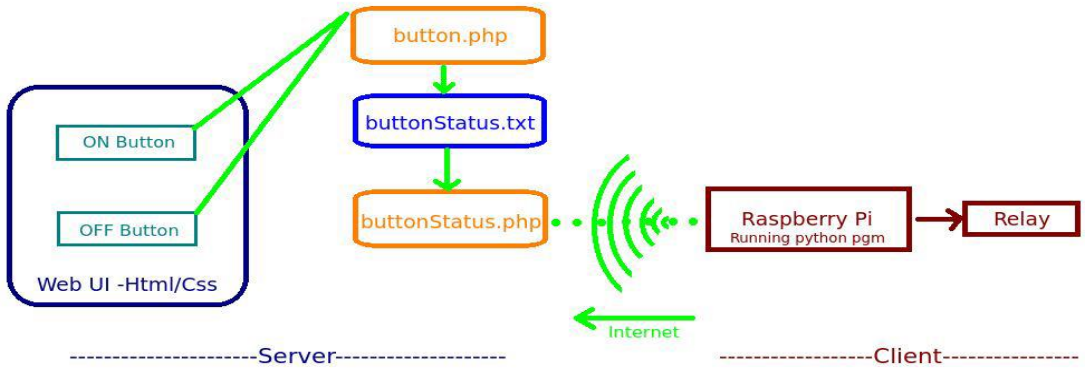


Fig. 7.2.1 Working

Thus the current button press state is recorded in the text file: buttonStatus.txt. The client side consists of a Raspberry Pi with a relay circuit connected to its GPIO pin. The pi runs a python program which is used to 'Post' a URL link using urllib3.

That is, the pi constantly reads the contents of a URL link. Here, the URL link is another PHP file called buttonStatus.php.

This PHP file serves as an API to read the contents of the text file buttonStatus.txt. After reading the data, the python program checks if the string obtained is “ON” / “OFF” based on which it switches ON/OFF the relay respectively via its GPIO pin.

7.3 Index.php

```
<html>


<head>

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.8/jquery.min.js">

</script>

<style>

html, body

{

   height: 100%;

   margin:0;

   padding:0;

}


div {

   position:relative;

   height: 100%;

   width:100%;

}


#backgrd {
```

```
    position:absolute;
    top:0;
    left:0;
    right:0;
    bottom:0;
    margin:auto;
    z-index:-1;
}
#image1{
display:block;
width:50px;
height:50px;
position:absolute;
top:37%;
left:40%;
right:50%;
bottom:50%;
}
#image2{
display:none;
width:50px;
height:50px;
position:absolute;
top:37%;
```

```
left:40%;
right:50%;
bottom:50%;
}
#f1{
display:block;
width:50px;
height:50px;
position:absolute;
left:51%;
top:50%;
right:50%;
botom:50%;

}
#f2{
display:none;
width:50px;
height:50px;
position:absolute;
left:51%;
top:50%;
right:50%;
botom:50%;
```

```
}


</style>
<script>
   <?php
   if($status=="0")
   {
   ?>
    document.getElementById("image1").style.display="block";
      document.getElementById("image2").style.display="none";


       <?php
   }else{
       ?>
           document.getElementById("image1").style.display="none";
      document.getElementById("image2").style.display="block";
           <?php
   }
           ?>
function change(id){
var x= document.getElementById("image1");
var y= document.getElementById("image2");
```

```
var value;
if(id=="image1")
{

x.style.display="none";
y.style.display="block";
value=1;
        var xhttp = new XMLHttpRequest();

   xhttp.open("GET", "light1.php?light1="+value, true);
   xhttp.send();
}
else{
x.style.display="block";
y.style.display="none";
value=0;
var xhttp = new XMLHttpRequest();

   xhttp.open("GET", "light1.php?light1="+value, true);
   xhttp.send();
}
}
function changeFan(id){
var x= document.getElementById("f1");
```

```
var y= document.getElementById("f2");
var value;
if(id=="f1")
{
x.style.display="none";
y.style.display="block";
value=1;
        var xhttp = new XMLHttpRequest();

    xhttp.open("GET", "fan.php?fan="+value, true);
    xhttp.send();
}
else{
x.style.display="block";
y.style.display="none";
value=0;
var xhttp = new XMLHttpRequest();

    xhttp.open("GET", "fan.php?fan="+value, true);
    xhttp.send();

}
}
```

```
</script>

</head>

<body>

<div><img src="http://cranberrycove.ca/wp-content/uploads/2017/02/welcome-img.png" id="backgrd" >

<img src="41divnkkBVL._SL500_AC_SS350_.jpg" id="image1" onclick="change(this.id)">

<img src="download.jpg" id="image2" onclick="change(this.id)">

<img src="images.png" id="f1" onclick="changeFan(this.id)"  />

<img src="giphy.gif" id="f2" onclick="changeFan(this.id)"  />

</div>

</body>


</html>
```

7.4 LightStatus.php

```
<?php

//$light1=$_REQUEST("light1");

$light1=filter_input(INPUT_GET, 'light1');

$myfile = fopen("light1.txt", "w") ;

fwrite($myfile, $light1);

fclose($myfile);

?>
```

7.5 FanStatus.php

```
<?php

//$light1=$_REQUEST("light1");

$fan=filter_input(INPUT_GET, 'fan');
```

```
$myfile = fopen("fan.txt", "w") ;

fwrite($myfile, $fan);

fclose($myfile);

?>
```

7.6 Light.py

```
import urllib3

import RPi._GPIO as GPIO

GPIO.setmode(GPIO.BOARD)

GPIO.setup(7,GPIO.OUT)

while 1:

        http=urllib3.PoolManager()

        r=http.request('GET','http://www.magichome.ml/light.txt')

        x=int(r.data)

        if (x==0):

                GPIO.output(7,False);

        else:

                GPIO.output(7,True);
```

7.7 Fan.py

```
import urllib3
import RPi._GPIO as GPIO
GPIO.setmode(GPIO.BOARD)
GPIO.setup(11,GPIO.OUT)
while 1:
        http=urllib3.PoolManager()
        r=http.request('GET','http://www.magichome.ml/fan.txt')
        x=int(r.data)
        if (x==0):
                GPIO.output(11,False);
        else:
                GPIO.output(11,True);
```

CHAPTER 8
TESTING

8.1 TESTING OBJECTIVES:

The main objective of testing is to uncover a host of errors, systematically and with minimum effort and time. Stating formally, we can say, testing is a process of executing a program with intent of finding an error. A successful test is one that uncovers an as yet undiscovered error. A good test case is one that has probability of finding an error, if it exists. The test is inadequate to detect possibly present errors. The software more or less confirms to the quality and reliable standards.

8.1.1 THE BASIC LEVELS OF TESTING:

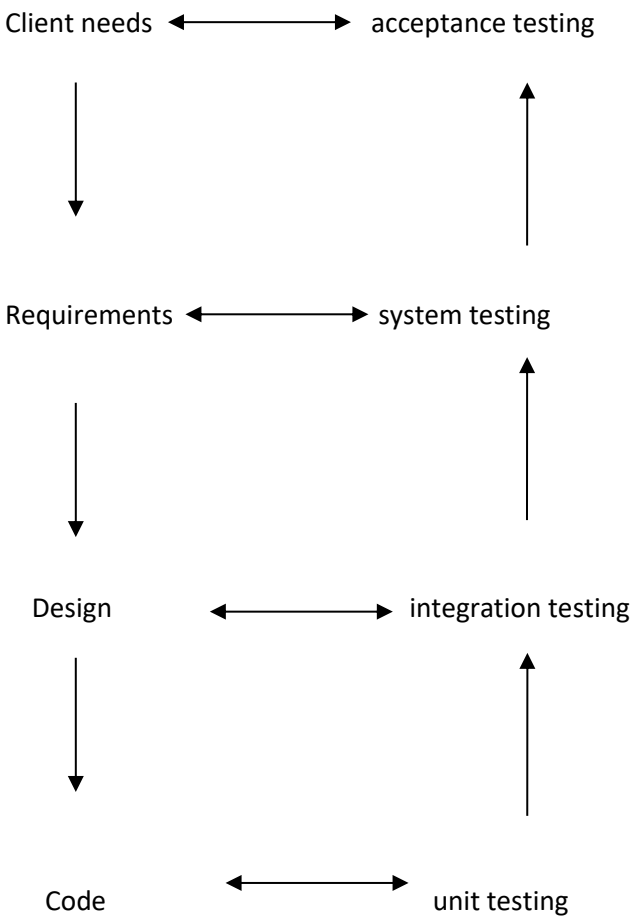


Figure 8.1.2.1. Levels of Testing

8.1.2 UNIT TESTING (White Box Testing):

At the lowest level, the function of the basic unit of software is tested in isolation. This is where the most detailed investigation of the internal workings of individual units are carried out. Unit testing is often performed by the programmer who wrote the code.

The purpose of unit testing is to find errors in the individual units, which could be data or logic related errors. The test can be derived from the program specification or design document. Units which cannot be tested in isolation may require the creation of small test programs known as test harness.

Activities to be followed for Unit Testing:

- Ensure that all paths are traversed and branching takes properly.
- Ensure that instructions related to a testcase are executed properly.
- Verify operation at normal value range.
- Verify operation outside range values.
- Verify program execution at boundary conditions.
- Check the calls to all programs.
- Ensure all data structures are handled properly.
- Check file handling.
- Ensure that all loops are terminated normally.
- Identify and remove abnormal termination of all loops.
- Ensure all errors are trapped.
- Check the values returned from called programs (stubs may be used if called program is not yet ready)

8.1.3 INTEGRATION TESTING (Interfaces Testing):

When two or more tested units are combined the tests should look for errors in two ways; in the interfaces between the units and in the functions which can be performed by the integrated unit which could not be assessed during unit testing.

8.1.4 SYSTEM TESTING (Functionality Testing):

After integration testing is completed the entire system is tested as whole. System testing looks for errors in the end-to-end functionality of the system and also for errors in non-functional quality attributes such as performance, reliability, volume, usability, maintainability security etc. System testing can be carried out by independent personnel. The tests are derived from the Functional Specification:

8.1.4.1 REGRESSION TESTING:

Regression testing is a testing process that is applied after the programs are modified. Regression testing is a major component in maintenance and conversion projects.

Modifying a program involves creating new logic to correct an error or implement a change, and incorporating that logic into an existing program. The new logic may involve minor modification such as adding, deleting or rewriting a few lines of code or may involve major modifications such as adding, deleting or replacing one or more modules or sub-systems. Regression testing aims to check the correctness of the new logic, to ensure the continuous working of the unmodified portions of a program and to validate that the whole functions correctly.

8.1.5 ACCEPTANCE TESTING (Black Box Testing):

After system testing or regression testing is completed, the system is handed over to the customer or user, and acceptance testing marks the transition from ownership by the developers to the ownership by the users. The acceptance test is different in nature in three ways:

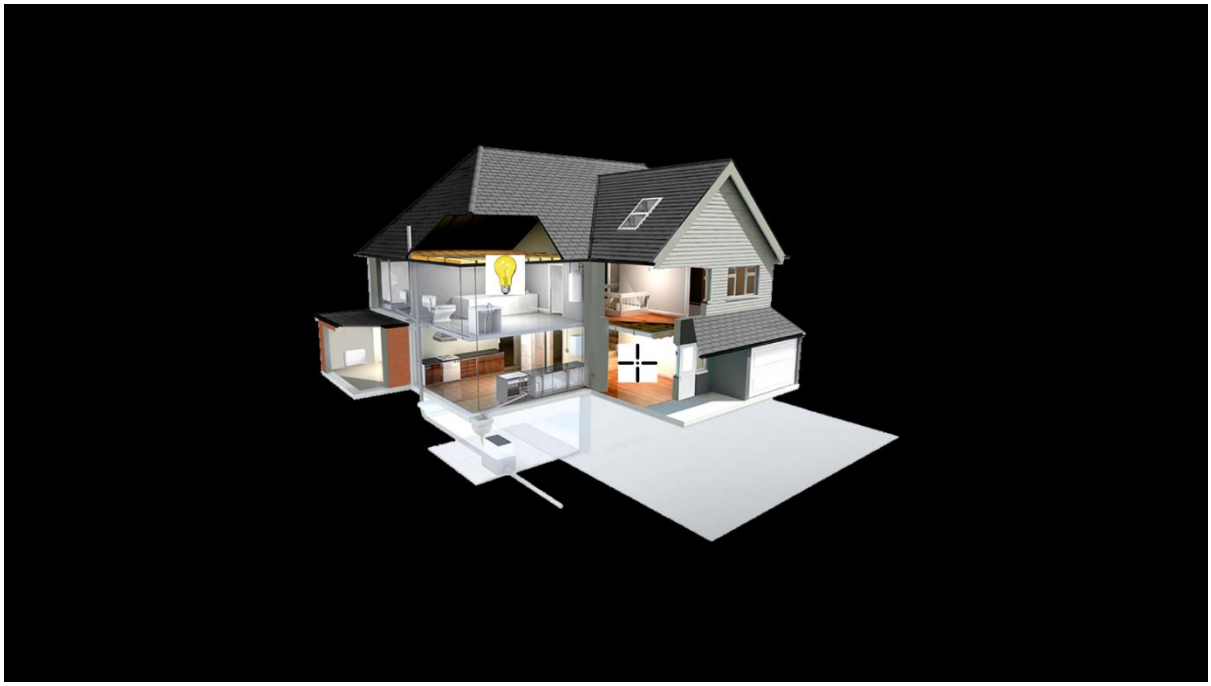
- It is the responsibility of the accepting organization rather than the developing organization
- The purpose of acceptance testing is to give confidence that the system is working rather than trying to find errors. The acceptance testing is a demonstration rather than a test.
- Acceptance testing also includes testing of the user organization working practices, to ensure the computer system will fit the operational procedures. The acceptance test gives a confidence to the users that the system is ready for operational use.

8.2 TEST PLANS

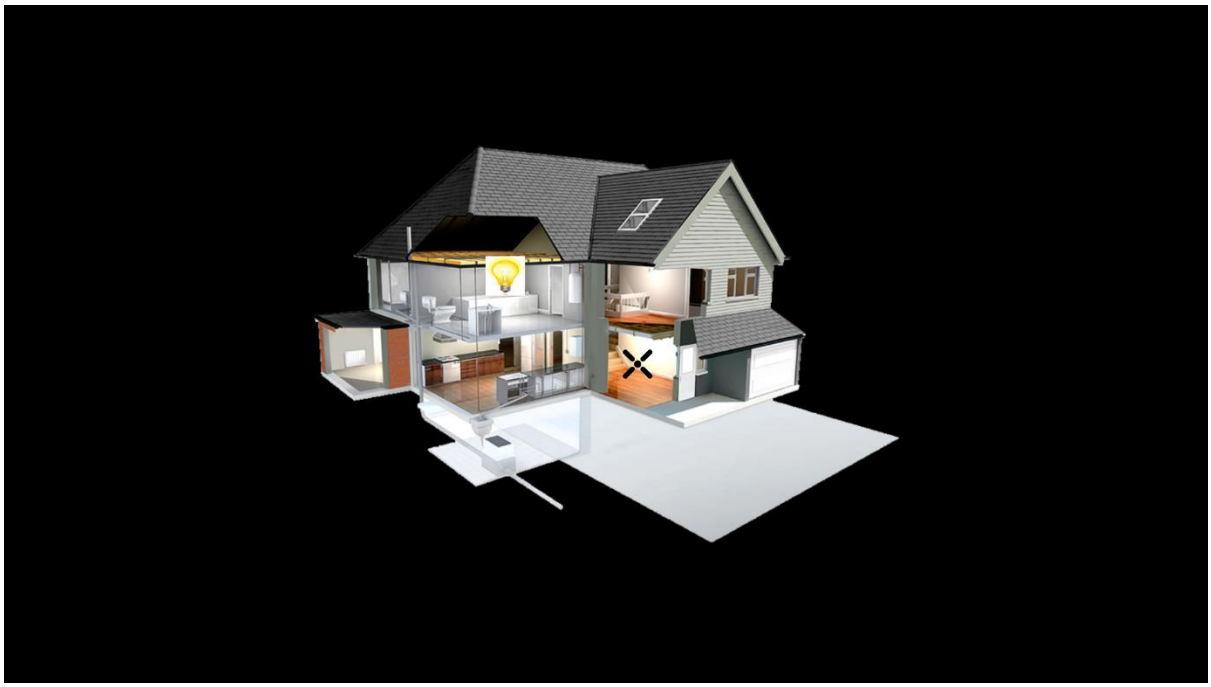
S.no	Test Case	Description	Outcome	Result
1	Switch on appliance using UI.	Light/fan should be switched on using the buttons designed in UI.	Appliance is turned on.	Passed.
2	Switch on appliance using one device and turn off using other device.	Light/fan should be turned on using UI Buttons from one device and turned off using UI Buttons from other device.	Appliance turned on and off based on button status.	Failed.
3	Manually turn on appliance and later turn off using UI buttons.	Switch on light/fan manually and turn off using UI buttons.	Appliance turned off.	Passed.

Table. 8.2.1 Test Cases for the system.

CHAPTER 9
SCREENSHOTS



Screenshot. 9.1. Home Page with Appliances OFF



Screenshot. 9.2. Home Page with Appliances ON

Fig. 9.3. Raspberry pi 2 model B

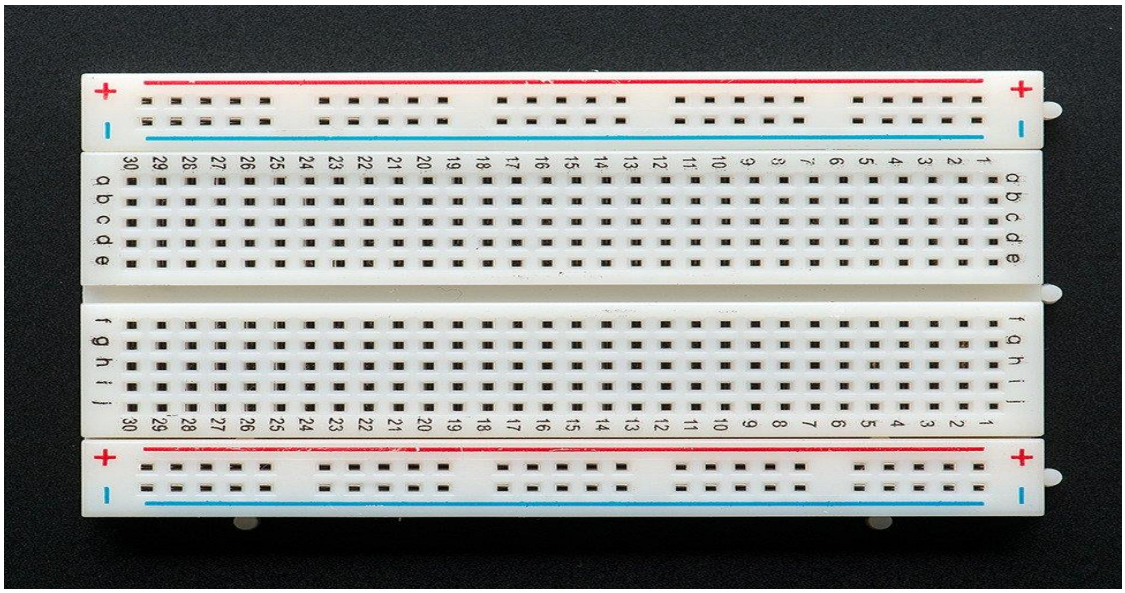


Fig. 9.4. Breadboard

Fig.9.5 Jumper cables

Fig 9.6 Led lights

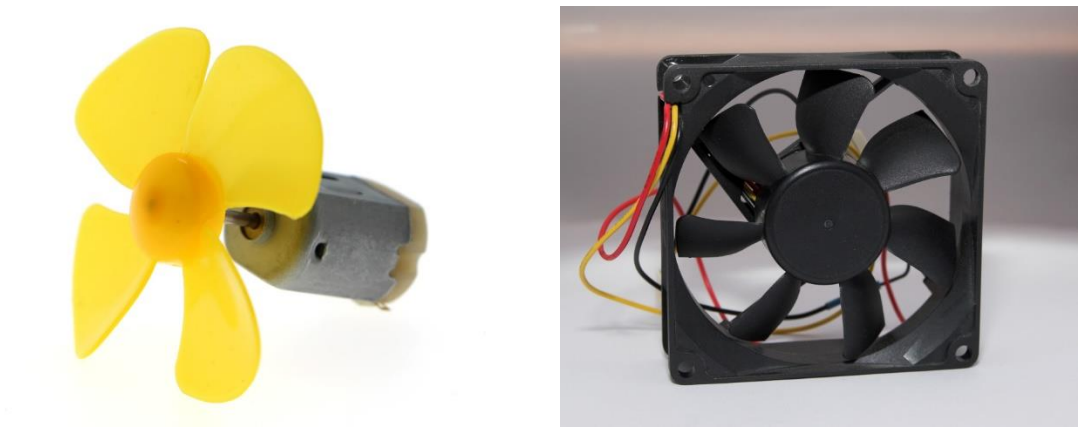


Fig 9.7 Fan

Fig 9.10 Relay

Fig 9.11 Wi-Fi Adapter

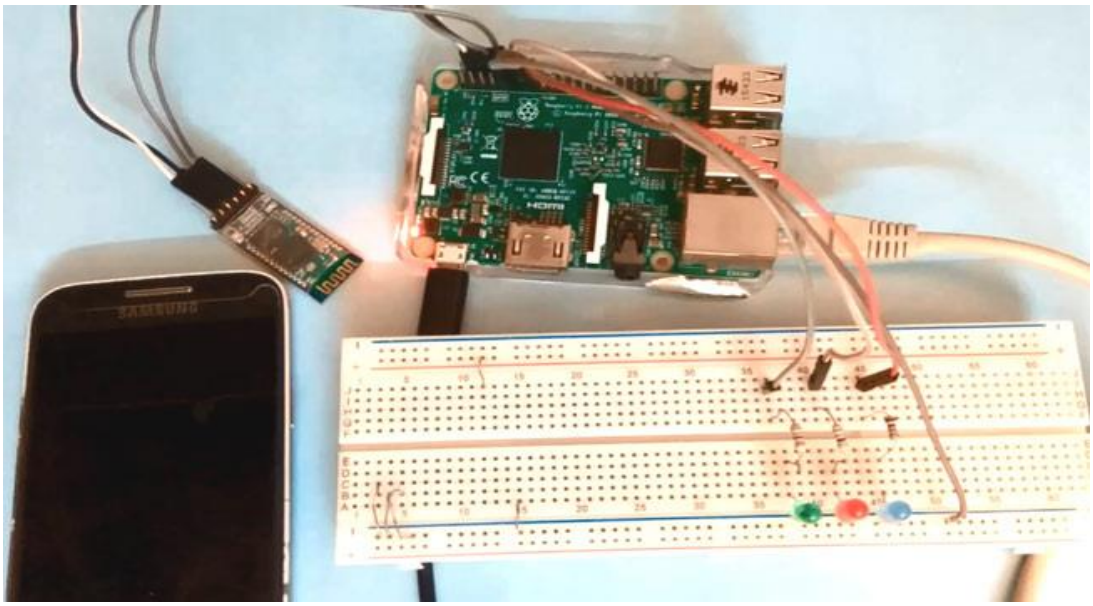


Fig 9.12 Complete Set-up

CONCLUSION

10.1 CONCLUSION:

The application of the IoT technology, in home automation means combination of all electrical devices like smart mobile phone, personal computer, tablet and their monitoring, controlling and alerting in ways not possible before. This proposed system provides many advantages including, safety, security, improved comfort, energy and cost savings. In order to address the issues of flexibility and functionality, a novel, standalone, flexible and low cost home controlling and monitoring system using Web services as an interoperable layer for communicating between the remotely present user and the home devices, have been designed. Performed research have shown that by using the Raspberry Pi and open source software it is possible to programmatically control many devices in a home in such a way that user can create his/her own solution customized to meet his/her individual needs. Thus, the proposed system is better from the scalability, flexibility and security point of view than the commercially available home automation systems.

10.2 FUTURE SCOPE

The future of Home Automation is very vast. We can add many more new features in it in future to make it more efficient to fulfil user's requirement. Some features are given below.

1. Home Security system goes wireless.
2. It can be operated through voice command
3. Temperature sensors
4. User can add new appliances by their own in a given interface without any external support.

BIBLIOGRAPHY

FOR INSTALLATION:

- **https://www.apachefriends.org/download.html**
- **https://www.python.org/downloads/**
- **http://php.net/downloads.php/**

REFERENCES:

- Raspberry Pi Home Automation Based on Internet of Things (IoT). Upadhye Madhuri Ganesh R. A. Khan, IJARCCE,2012.
- Web-based Raspberry Pi home automation system using normal HTTP protocols. Arvind Sanjeev, Maker pro.
- RASPBERRY PI HOME AUTOMATION. P. Bhaskar Rao, S. K. Uma, IJCSMC. 2015.
- IOT Based Home Automation Using Raspberry Pi. Odunlade Emmanuel,CircuitDigest.