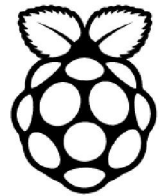


## บทที่ 12

# Raspberry Pi 2 กับการควบคุม พอร์ตอินพุตเอาต์พุตผ่านเครือข่าย



ด้วยความสามารถของบอร์ด Raspberry Pi 2 ที่รองรับการเชื่อมต่อเครือข่ายได้สะดวกเหมือนกับคอมพิวเตอร์เครื่องหนึ่ง ดังนั้นจึงมีการนำ Raspberry Pi 2 มาใช้ในการควบคุมหรือติดต่ออุปกรณ์อินพุตเอาต์พุตผ่านเครือข่ายอย่างกว้างขวาง ทั้งในแบบเครือข่ายภายในองค์กร และผ่านเครือข่ายอินเทอร์เน็ต หากแต่การนำเสนอขั้นตอนการพัฒนาบอร์ด Raspberry Pi 2 เพื่อใช้ประโยชน์ในลักษณะกลับเป็นไปอย่างจำกัด ดังนั้นในบทนี้จึงเป็นการนำเสนอตัวอย่างขั้นตอนการพัฒนาและใช้งานบอร์ด Raspberry Pi 2 ในการติดต่อกับอุปกรณ์อินพุตเอาต์พุตผ่านเครือข่ายข้อมูล โดยในการอธิบายนั้นจะมีการยกตัวอย่าง ประกอบกับโค้ดทั้งภาษา HTML (Hyper Text Markup Language) และ Python เพื่อนำไปสู่การสร้างแอปพลิเคชัน Simple I/O Web Server ตั้งแต่ตัวอย่างการควบคุมอุปกรณ์เอาต์พุตผ่านเครือข่าย และการรับค่าจากอุปกรณ์อินพุตรวมถึงผ่านข้อมูลจากตัวตรวจจับเพื่อนำมาแสดงผลบนเครือข่าย โดยบอร์ด Raspberry Pi 2 จะทำหน้าที่เป็นเว็บเซิร์ฟเวอร์หลัก

เนื้อหาในบทนี้จะมีรายละเอียดอยู่มากพอสมควร รวมถึงในขั้นตอนการทดลองปฏิบัติการด้วยจะมีการยกตัวอย่างเพื่อให้เห็นถึงที่มาที่ไปของส่วนประกอบต่างๆ ที่จะนำไปสู่การสร้างแอปพลิเคชันเว็บเซิร์ฟเวอร์ควบคุมอุปกรณ์อินพุตเอาต์พุตจนสำเร็จ จึงขอให้ผู้สนใจโปรดอ่านและติดตามทำความเข้าใจตามลำดับหัวข้อ ตามขั้นตอน โดยสรุปหัวข้อต่างๆ ที่สำคัญของบทนี้ได้ดังนี้

12.1 กลไกการทำงานของเว็บเซิร์ฟเวอร์ในการควบคุมอุปกรณ์อินพุตเอาต์พุต

12.2 กลไกการทำงานระหว่างเว็บไคลเอนต์และเว็บเซิร์ฟเวอร์

12.3 เตรียมอุปกรณ์เพื่อการพัฒนา Raspberry PI 2 สำหรับงานควบคุมผ่านเครือข่าย

12.4 การทำให้บอร์ด Raspberry Pi 2 เป็นเว็บเซิร์ฟเวอร์ - จะได้เรียนรู้ขั้นตอนการทำให้บอร์ด Raspberry Pi 2 ทำงานเป็นเว็บเซิร์ฟเวอร์ได้ จะต้องมีการติดตั้งซอฟต์แวร์เพิ่มเติม ทั้ง Apache HTTP Server และ PHP

12.5 มารู้จักกับโครงสร้างของภาษา HTML - เมื่อต้องการทำเว็บเซิร์ฟเวอร์ สิ่งหนึ่งที่ต้องทำให้ได้คือ การเขียนเว็บเพจอย่างง่ายด้วยภาษา HTML การทำความเข้าใจกับโครงสร้างของภาษานี้จึงเป็นสิ่งจำเป็น

12.6 การสร้างเว็บเพจบนบอร์ด Raspberry Pi 2 - ต่อยอดความรู้จากหัวข้อ 12.4 มาเขียนโปรแกรมเพื่อสร้างเว็บเพจของตัวเว็บเซิร์ฟเวอร์ซึ่งรันบนบอร์ด Raspberry Pi 2

12.7 การใช้งาน PHP - เป็นอีกหนึ่งภาษาคอมพิวเตอร์ที่นิยมใช้มากในการสร้างเว็บเพจและทำเว็บเซิร์ฟเวอร์

12.8 สร้างเว็บเพจด้วย PHP - จากการเรียนรู้ในหัวข้อ 12.6 นำข้อมูลและตัวอย่างโปรแกรมต่างๆ มาสร้างเว็บเพจด้วยภาษา PHP

12.9 คำสั่งพื้นฐานในการควบคุมขาพอร์ต GPIO แบบคอมมานด์ไลน์ - ในหัวข้อนี้แนะนำการเขียนโปรแกรมสั้นๆ ในแบบคอมมานด์ไลน์เพื่อติดต่อและควบคุมขา GPIO ได้รับการนำไปใช้เมื่อต้องการสั่งงาน GPIO จากเว็บไคลเอ็นต์ผ่านเซิร์ฟเวอร์

12.10 การรับส่งข้อมูลระหว่างเว็บเพจ

12.11 การสั่งให้สคริปต์ Python ทำงานด้วย PHP

12.12 การทดลองควบคุมอุปกรณ์เอาต์พุตอย่างง่ายผ่านเว็บเบราว์เซอร์โดยใช้ PHP และ HTML

12.13 การทดลองติดต่ออุปกรณ์อินพุตเอาต์พุตอย่างง่ายผ่านเว็บเบราว์เซอร์ทำงานร่วมกับไฟล์สคริปต์ Python

## 12.1 กลไกการทำงานของเว็บเซิร์ฟเวอร์

### ในการควบคุมอุปกรณ์อินพุตเอาต์พุต

เมื่อนำบอร์ด Raspberry Pi 2 มาทำหน้าที่เป็นเว็บเซิร์ฟเวอร์ การทำงานจะแบ่งออกเป็น 2 ส่วนคือ

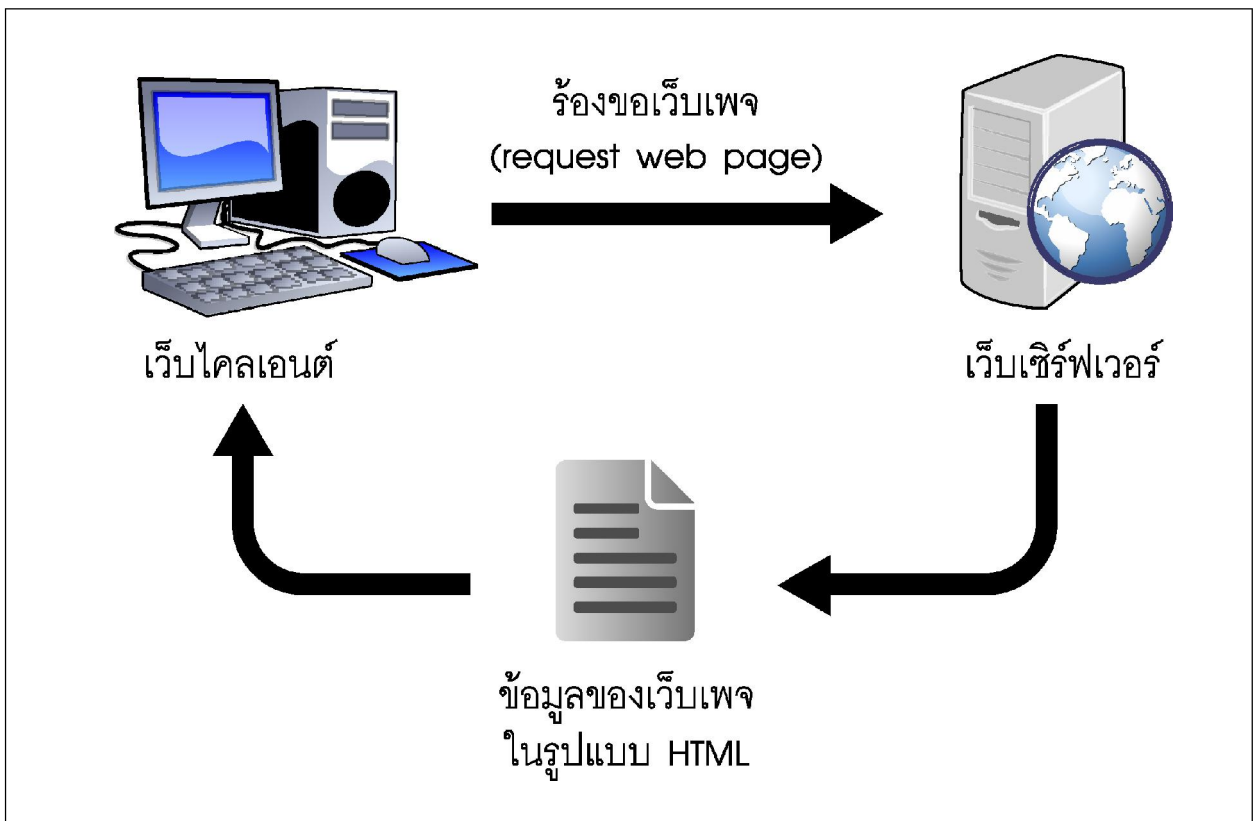
1. ส่วนของการแสดงผลหน้าเว็บ ใช้การสร้างหน้าเว็บหรือเว็บเพจ (web page) ด้วยโปรแกรมภาษา HTML

2. ส่วนของโปรแกรมที่ติดต่อกับอุปกรณ์อินพุตเอาต์พุตของบอร์ด Raspberry Pi 2 ซึ่งต้องทำงานควบคู่กันกับเว็บเพจ

การทำงานของบอร์ด Raspberry Pi 2 ที่ต้องกระทำควบคู่กับเว็บเพจ ในที่นี้ขอนำเสนอ 2 แบบประกอบด้วย

1. กำหนดให้โปรแกรมส่วนที่ติดต่อกับอุปกรณ์อินพุตเอาต์พุตแทรกอยู่ภายในส่วนที่ใช้แสดงเว็บเพจ หากเขียนเว็บเพจด้วย PHP ก็จะใช้แทรกโปรแกรมส่วนที่ติดต่อกับอุปกรณ์อินพุตเอาต์พุตเข้าไปด้วย รูปแบบนี้ใช้ได้ดีกับการทำงานเพียงรอบเดียวจบ ใช้เวลาน้อยในการทำงาน เพราะเว็บเพจจะไม่เปลี่ยนแปลง จนกว่าจะกระทำทุกคำสั่งจนเสร็จ จะแสดงการทำงานในลักษณะนี้ในตัวอย่างการทดลองควบคุมอุปกรณ์ ผ่านเว็บเบราว์เซอร์โดยใช้ PHP และ HTML (หัวข้อ 12.12 ในบทนี้)

2. กำหนดให้ใช้ข้อมูลร่วมกันระหว่างโปรแกรมที่ติดต่อกับอุปกรณ์อินพุตเอาต์พุตกับโปรแกรมที่ใช้แสดงเว็บเพจ เช่น กำหนดให้โปรแกรมในส่วนที่ติดต่อกับอุปกรณ์อินพุตเอาต์พุตทำงานตลอดเวลา โดยมีการนำข้อมูลบางส่วนที่โปรแกรมของการแสดงเว็บเพจก็ใช้งานอยู่ด้วยมาประกอบเพื่อให้การทำงานสมบูรณ์ ข้อมูลที่ใช้ร่วมกันอาจอยู่ในรูปของไฟล์เอกสารหรือฐานข้อมูลก็ได้ ด้วยวิธีการนี้จะยืดหยุ่นกว่าแบบแรก เพราะการทำงานของโปรแกรมทั้งสองแยกออกจากกันได้อย่างชัดเจน มีเพียงการใช้งานข้อมูลบางส่วนร่วมกันเท่านั้น ถึงแม้ว่าโปรแกรมของการติดต่อกับอุปกรณ์อินพุตเอาต์พุตอาจมีการประมวลผลที่ซับซ้อนและใช้เวลานาน โปรแกรมของการแสดงเว็บเพจก็ยังคงทำงานไปได้อย่างต่อเนื่อง เพราะใช้ข้อมูลจากไฟล์เอกสารหรือฐานข้อมูลที่มีอยู่ในปัจจุบัน เพียงแต่ข้อมูลที่นำมาแสดงผลนั้นอาจยังไม่ได้รับการปรับปรุงจากการทำงานล่าสุดของโปรแกรมที่ใช้ติดต่อกับอุปกรณ์อินพุตเอาต์พุต (เพราะอาจยังประมวลผลไม่เสร็จ) การทำงานในลักษณะนี้อธิบายให้เข้าใจได้เพิ่มขึ้นด้วยตัวอย่างการทดลองควบคุมอุปกรณ์ผ่านเว็บเบราว์เซอร์ร่วมกับไฟล์สคริปต์ Python ในหัวข้อ 12.13 ของบทนี้



รูปที่ 12-1 ไดอะแกรมแสดงกลไกการทำงานร่วมกันของเว็บไคลเอนต์ (ปกติคือ คอมพิวเตอร์ของผู้ใช้งาน) และเว็บเซิร์ฟเวอร์ โดยทำงานผ่านเว็บเบราว์เซอร์เพื่อแจ้งร้องขอเว็บเพจ และรับข้อมูลกลับมาในรูปแบบ HTML เพื่อนำมาแสดงเว็บเพจอย่างสมบูรณ์ที่เว็บไคลเอนต์

## 12.2 กลไกการทำงานระหว่างเว็บไคลเอนต์และเว็บเซิร์ฟเวอร์

แสดงด้วยไดอะแกรมในรูปที่ 12-1 เริ่มต้นเมื่อเว็บไคลเอนต์ (web client) ซึ่งก็คือ คอมพิวเตอร์ของผู้ใช้งานทำการร้องขอเว็บเพจ (Request Web Page) มายังเว็บเซิร์ฟเวอร์ (web server) จากนั้นเว็บเซิร์ฟเวอร์จะส่งข้อมูลเว็บเพจที่ร้องขอมายังเว็บไคลเอนต์ ในรูปแบบ HTML แล้วแสดงผลด้วยเว็บเบราว์เซอร์ที่ติดตั้งอยู่ในเว็บไคลเอนต์

เนื่องจากเว็บเพจที่ถูกสร้างด้วยภาษา HTML เป็นภาษาที่ใช้แสดงหน้าตาเว็บเพจตามที่ต้องการแต่จะไม่ช่วยให้เว็บเพจนั้นปรับเปลี่ยนหน้าตาได้เอง การสร้างเว็บเพจที่สามารถปรับเปลี่ยนได้นั้นทำได้หลายวิธี หนึ่งในนั้นก็คือ การฝังสคริปต์หรือชุดคำสั่งที่ทำงานทางฝั่งเซิร์ฟเวอร์ (server-side script)-ไว้ในเว็บเพจ สคริปต์ตัวหนึ่งที่นิยมใช้กันมากจากคือ PHP engine (PHP: Professional Home Page) เป็นสคริปต์ที่ทำงานร่วมกับภาษา HTML เพื่อช่วยให้เว็บเพจเปลี่ยนหน้าเว็บได้เองหรือเรียกอีกอย่างหนึ่งคือ ทำให้เว็บเพจมี “ความฉลาด” ขึ้น

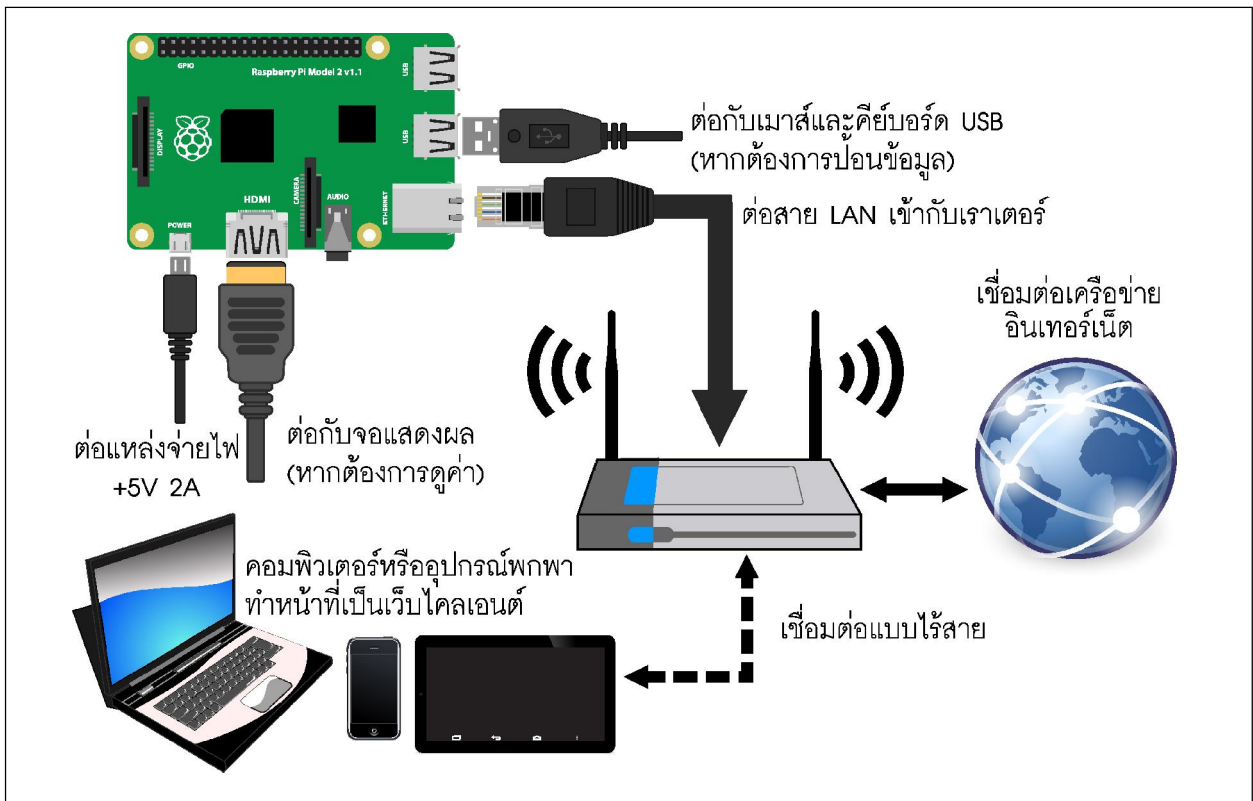
## 12.3 การเตรียมอุปกรณ์เพื่อการพัฒนา Raspberry Pi 2

### สำหรับงานควบคุมผ่านเครือข่าย

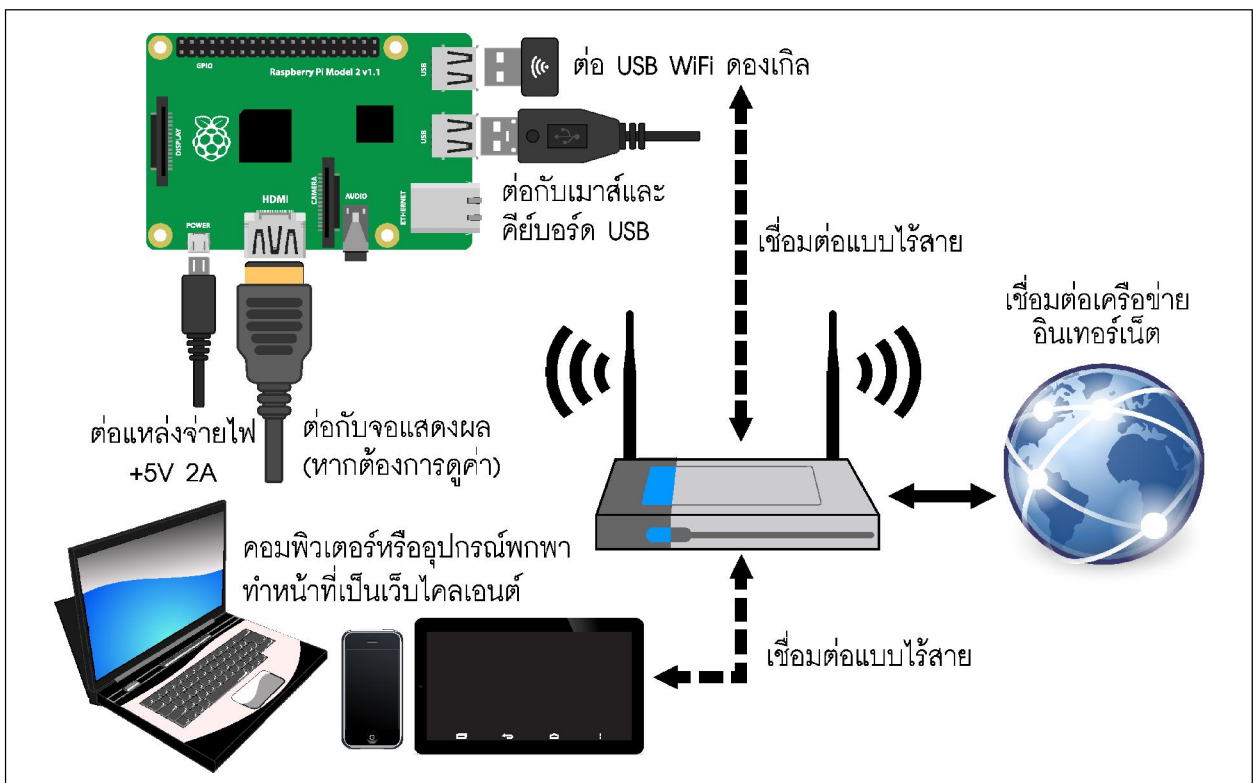
เพื่อให้การเรียนรู้เพื่อพัฒนาบอร์ดคอมพิวเตอร์อย่าง Raspberry Pi 2 สำหรับการสร้างระบบควบคุมผ่านเครือข่าย การจัดเตรียมเครื่องมือและอุปกรณ์เป็นสิ่งที่ต้องดำเนินการไปพร้อมกัน อันประกอบด้วย

1. บอร์ด Raspberry Pi 2
2. USB WiFi ดองเกิล (มีให้ในชุด Raspberry Pi 2 Tech Kit แล้ว)
3. เราเตอร์แบบไร้สาย (Wireless router) หรือแบบปกติก็ได้ (หากใช้แบบปกติ ไม่ต้องใช้ USB WiFi ดองเกิลตาม ไดอะแกรมในรูปที่ 12-3) พร้อมแหล่งจ่ายไฟ
4. สาย LAN เพื่อเชื่อมต่อบอร์ด Raspberry Pi 2 เข้ากับเราเตอร์ (กรณีไม่ใช่แบบไร้สายตามไดอะแกรมในรูปที่ 12-3)
5. อะแดปเตอร์ +5V ที่จ่ายกระแสไฟฟ้าได้ 2A เป็นอย่างน้อย สำหรับบอร์ด Raspberry Pi 2 (มีให้ในชุด Raspberry Pi 2 Tech Kit แล้ว)
6. คอมพิวเตอร์ที่เชื่อมต่อเข้ากับระบบเครือข่าย
7. เครือข่ายอินเทอร์เน็ต (ในกรณีต้องการเชื่อมต่อกับเครือข่ายภายนอก)

แนวทางการเชื่อมต่ออุปกรณ์ทั้งหมด เพื่อการเรียนรู้และทดลองพัฒนาระบบควบคุมผ่านเครือข่ายของบอร์ด Raspberry Pi 2 แสดงในรูปที่ 12-2



รูปที่ 12-2 ไดอะแกรมการเชื่อมต่ออุปกรณ์เพื่อพัฒนาระบบควบคุมผ่านเครือข่ายโดยใช้บอร์ด Raspberry Pi 2 ในแบบต่อสาย LAN



รูปที่ 12-3 ไดอะแกรมการเชื่อมต่ออุปกรณ์เพื่อพัฒนาระบบควบคุมผ่านเครือข่ายโดยใช้บอร์ด Raspberry Pi 2 ในแบบไร้สายผ่าน WiFi

## 12.4 การทำให้บอร์ด Raspberry Pi 2 เป็นเว็บเซิร์ฟเวอร์

การกำหนดให้ Raspberry Pi 2 ทำหน้าที่เป็นเว็บเซิร์ฟเวอร์ ลำดับแรกต้องติดตั้ง โปรแกรมสำหรับการทำงานเป็นเว็บเซิร์ฟเวอร์ก่อน เนื่องจากระบบปฏิบัติการหลักของ Raspberry Pi 2 เป็น Linux ดังนั้น โปรแกรมเว็บเซิร์ฟเวอร์ที่แนะนำให้ติดตั้งคือ **Apache HTTP Server** โปรแกรมนี้มีหน้าที่บริการข้อมูล และติดตั้งกลไก PHP แก่เว็บ ไคลเอนต์ (คอมพิวเตอร์ย่อยในเครือข่าย) ที่ทำการร้องขอเข้ามา

### 12.4.1 ขั้นตอนการติดตั้ง Apache HTTP Server

(1) เชื่อมต่อ Raspberry Pi 2 เข้ากับเครือข่ายอินเทอร์เน็ต (ดูไดอะแกรมจากรูปที่ 12-2 หรือ 12-3 หรือจากบทที่ 2)

(2) เข้าสู่คอมพิวเตอร์พร้อมท์ แล้วพิมพ์คำสั่งเพื่อติดตั้ง โปรแกรม Apache HTTP Server

```
sudo apt-get install apache2 -y
```

```
pi@raspberrypi ~ $ sudo apt-get install apache2 -y
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following extra packages will be installed:
  apache2-mpm-worker apache2-utils apache2.2-bin apache2.2-common libapr1 libaprutil1
  libaprutil1-dbd-sqlite3 libaprutil1-ldap ssl-cert
Suggested packages:
  apache2-doc apache2-suexec apache2-suexec-custom openssl-blacklist
The following NEW packages will be installed:
  apache2 apache2-mpm-worker apache2-utils apache2.2-bin apache2.2-common libapr1 libaprutil1
  libaprutil1-dbd-sqlite3 libaprutil1-ldap ssl-cert
0 upgraded, 10 newly installed, 0 to remove and 113 not upgraded.
Need to get 1,355 kB of archives.
After this operation, 4,929 kB of additional disk space will be used.
```

(3) เมื่อทำการติดตั้งเสร็จเรียบร้อยแล้ว ทำการเปิดเว็บเบราว์เซอร์ (อาทิ Chromium หรือ Midori) ที่ช่องค้นหา ให้พิมพ์ **Localhost** หรือพิมพ์แอดเดรส **127.0.0.1** หรือ IP แอดเดรสของบอร์ด Raspberry Pi 2 เช่น **192.168.1.36** จะแสดงข้อความที่ขึ้นต้นว่า **It works!** ดังรูป แสดงว่า Apache ทำงานได้ปกติ (การค้นหามหาเลข IP แอดเดรสของ Raspberry Pi 2 ที่อยู่ในเครือข่าย อาจใช้โปรแกรม Advance IP Scanner ช่วย ดูรายละเอียดได้จากกรอบแยกในบทนี้)

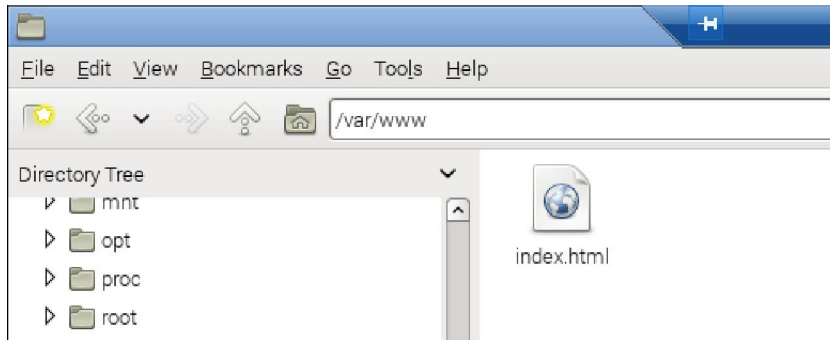


**It works!**

This is the default web page for this server.

The web server software is running but no content has been added, yet.

(4) เมื่อติดตั้งเสร็จแล้ว บน Raspberry Pi 2 จะมีไดเรกทอรีหรือโฟลเดอร์ที่ชื่อว่า **www** ถูกสร้างขึ้นในไดเรกทอรี **var** เป็นที่เก็บไฟล์สำหรับทำเป็นเว็บเพจหรือหน้าเว็บ เมื่อเปิดเข้าไปในไดเรกทอรี **/var/www** จะมีไฟล์ **index.html** อยู่ข้างใน ดังรูป



(5) ถ้าต้องการตั้งค่าเพื่อกำหนดระดับของการเข้าถึงหรือ permission ให้กับไดเรกทอรี **/var/www** เพื่อให้การส่งไฟล์ระหว่างคอมพิวเตอร์ทำได้สะดวกและง่ายต่อการพัฒนา ให้พิมพ์คำสั่งบน Raspberry Pi 2 ดังนี้

```
sudo chmod 0777 -R /var/
```

ตัวเลข 777 เป็นการตั้งค่าให้ผู้ใช้งานในทุกะดับการเข้าถึงสามารถอ่าน, เขียน และสั่งให้ทำงาน (Read-Write-Execute) ได้ทุกไฟล์และโฟลเดอร์ (หรือไดเรกทอรี)

## 12.4.2 ติดตั้ง PHP

(1) เชื่อมต่อ Raspberry Pi 2 เข้ากับเครือข่ายอินเทอร์เน็ต (ดูไดอะแกรมจากรูปที่ 12-2 หรือ 12-3 หรือจากบทที่ 2)

(2) เข้าสู่หน้าต่างเทอร์มินอล แล้วพิมพ์คำสั่งเพื่อติดตั้ง PHP

```
sudo apt-get install php5 libapache2-mod-php5 -y
```

```
pi@raspberrypi ~ $ sudo apt-get install php5 libapache2-mod-php5 -y
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following extra packages will be installed:
  apache2-mpm-prefork libonig2 libqdbm14 lsof php5-cli php5-common
Suggested packages:
  php-pear
The following packages will be REMOVED:
  apache2-mpm-worker
The following NEW packages will be installed:
  apache2-mpm-prefork libapache2-mod-php5 libonig2 libqdbm14 lsof php5
  php5-cli php5-common
0 upgraded, 8 newly installed, 1 to remove and 113 not upgraded.
```



## Advanced IP Scanner

# ซอฟต์แวร์ตรวจสอบหมายเลข IP แอดเดรสของอุปกรณ์ในระบบเครือข่าย

การค้นหาหมายเลข IP แอดเดรสของอุปกรณ์ในเครือข่าย จะต้องทราบขอบเขตของจำนวน IP แอดเดรสก่อนจึงจะค้นหาหมายเลข IP แอดเดรสนั้นได้ ดังนั้นอุปกรณ์ที่ต้องการทราบหมายเลข IP แอดเดรสจะต้องอยู่ใน LAN หรือเครือข่ายเดียวกัน ในการทดลองเพื่อเรียนรู้การใช้งานบอร์ด Raspberry Pi 2 ในการควบคุมอุปกรณ์ผ่านเครือข่ายในบทนี้ จะต้องมีการตรวจสอบหมายเลข IP แอดเดรสของบอร์ด Raspberry Pi 2 ที่เชื่อมต่อในเครือข่าย ดังนั้นจึงต้องมีการเชื่อมต่ออุปกรณ์ตามแนวทางในรูปที่ 12-2 หรือ 12-3 เสียก่อน จากนั้นใช้คอมพิวเตอร์ที่ทำหน้าที่เป็นเว็บไคลเอนต์ในการติดตั้งซอฟต์แวร์ Advanced IP Scanner เพื่อใช้ค้นหาหมายเลข IP แอดเดรสของบอร์ด Raspberry Pi 2 ต่อไป

## ติดตั้งโปรแกรม

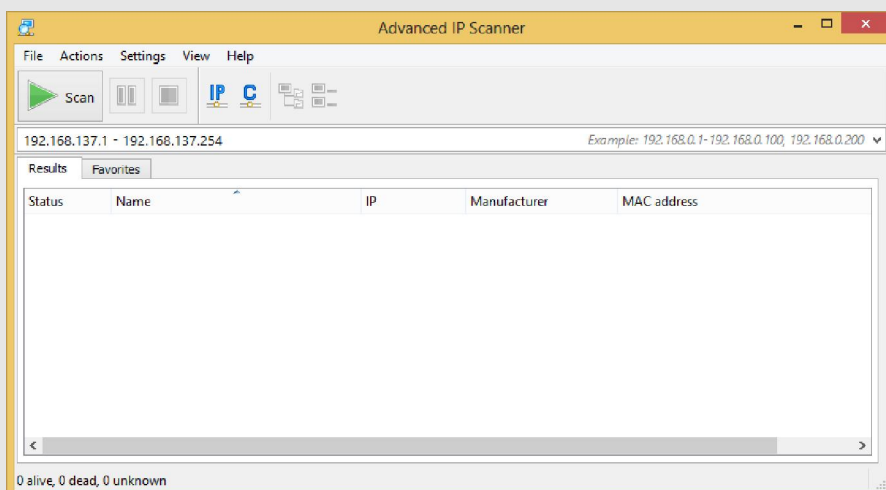
ซอฟต์แวร์นี้รองรับการสแกน HTTP, HTTPS, FTP และสแกนเครือข่ายเพื่อทราบข้อมูลเพิ่มเติมเกี่ยวกับอุปกรณ์ที่เชื่อมต่อกัน ได้แก่ ชื่อของคอมพิวเตอร์หรืออุปกรณ์ และ MAC แอดเดรส ดาวน์โหลดโปรแกรมได้ที่ <http://www.advanced-ip-scanner.com/th/> จากนั้นทำการติดตั้งโปรแกรมลงในคอมพิวเตอร์ (ที่ทำหน้าที่เป็นเว็บไคลเอนต์จากรูปที่ 12-2 หรือ 12-3) ให้เรียบร้อย

## การใช้งาน

- (1) เมื่อติดตั้งโปรแกรมเสร็จแล้ว จะมีไอคอนของซอฟต์แวร์ ดับเบิลคลิกเพื่อเปิดโปรแกรม

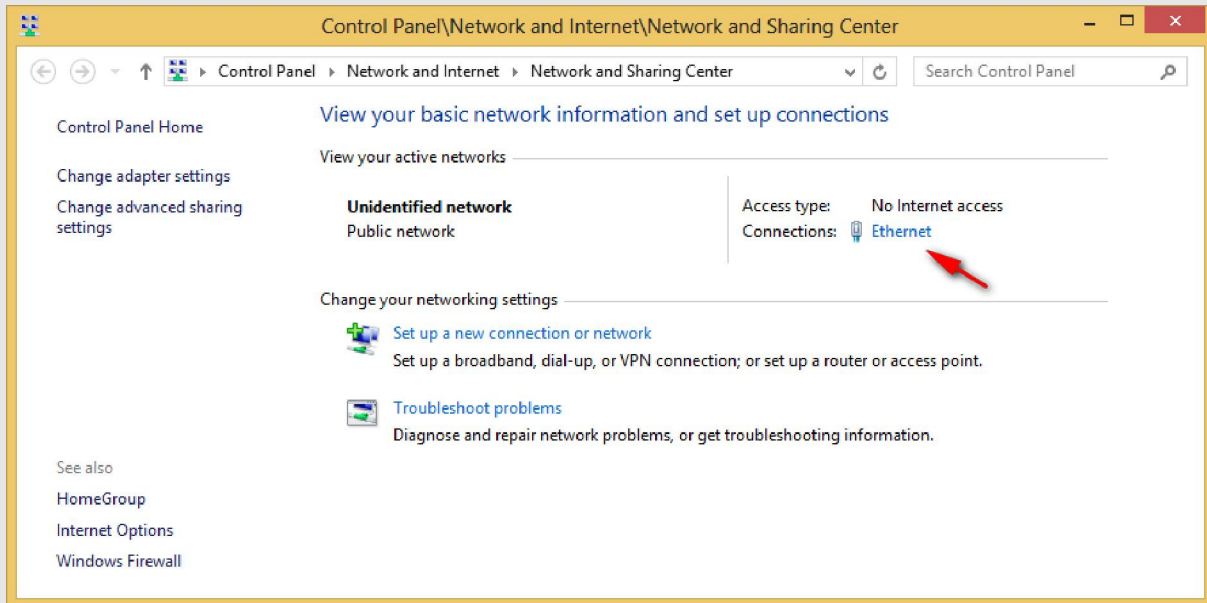


- (2) หน้าต่างหลักของโปรแกรมแสดงขึ้นมา มีรายละเอียดดังรูป

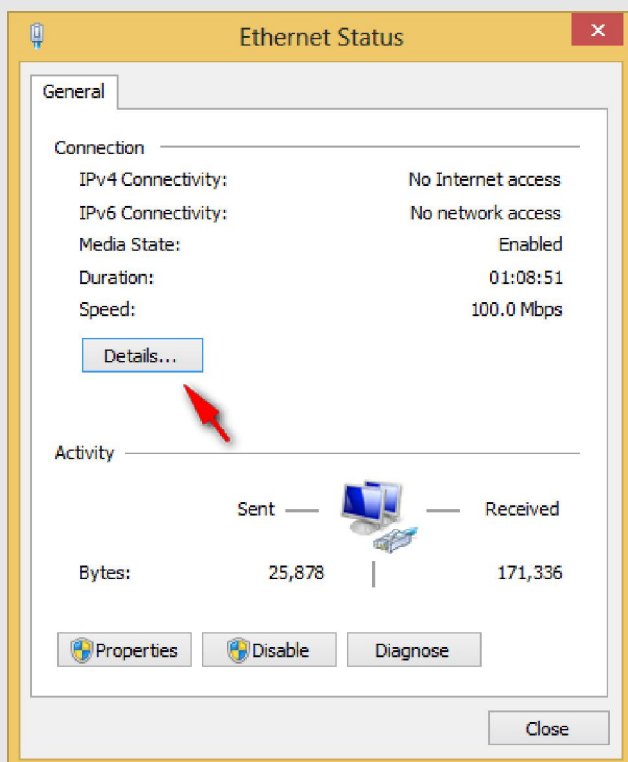




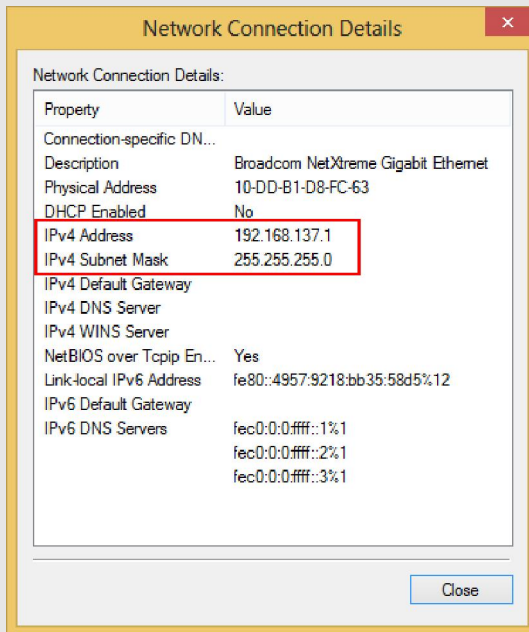
(3) วิธีการหาขอบเขต IP แอดเดรส ด้วยซอฟต์แวร์ Advanced IP Scanner เริ่มต้นด้วยการเปิด Control Panel > Network and Internet > Network and Sharing Center จากนั้นคลิกที่ Ethernet หากการต่ออุปกรณ์ใช้สาย LAN ต่อระหว่างคอมพิวเตอร์เราเตอร์ และ Raspberry Pi 2 เพื่อให้อยู่ในวง LAN เดียวกัน



- (4) แต่ถ้าหากบอร์ด Raspberry Pi 2 ต่อกับเครือข่ายในแบบไร้สายผ่าน Wi-Fi จะต้องคลิกที่ WiFi แทน
- (5) จากนั้นจะปรากฏหน้าต่าง Ethernet Status ที่แสดงสถานะของ Ethernet ให้คลิกที่ Details..



(6) จะปรากฏหน้าต่างแสดงรายละเอียดของการเชื่อมต่อ Network ดังรูป



ข้อมูลที่สำคัญสำหรับการหาขอบเขต IP แอดเดรสคือ IPv4 Address และ IPv4 Subnet Mask โดยที่

IPv4 Address คือ หมายเลข IP แอดเดรสของคอมพิวเตอร์ที่ได้รับหมายเลข IP แอดเดรสจากเราเตอร์

IPv4 Subnet Mask คือ จำนวนหมายเลข IP แอดเดรสของอุปกรณ์ที่อยู่ในวง LAN เดียวกัน นำข้อมูลนี้มาคำนวณขนาดของ IP แอดเดรส ดังนี้

(6.1) คำนวณหาหมายเลขเริ่มต้นของ IP แอดเดรสหรือ Network IP

IP แอดเดรส มีข้อมูล 4 ชุด ชุดละ 8 บิต คั่นด้วยจุด . การกระทำทางด้านลอจิกจะต้องกระทำให้ตรงชุดด้วย

IPv4 Address = 192.168.137.1

เมื่อแปลงเป็นเลขฐานสองจะได้ 11000000.10101000.10001001.00000001

IPv4 Subnet Mask=255.255.255.0

แปลงเป็นเลขฐานสองจะได้ 11111111.11111111.11111111.00000000

นำ IPv4 Address แอนด์ (AND) กับ IPv4 Subnet Mask จะได้

11000000.10101000.10001001.00000000

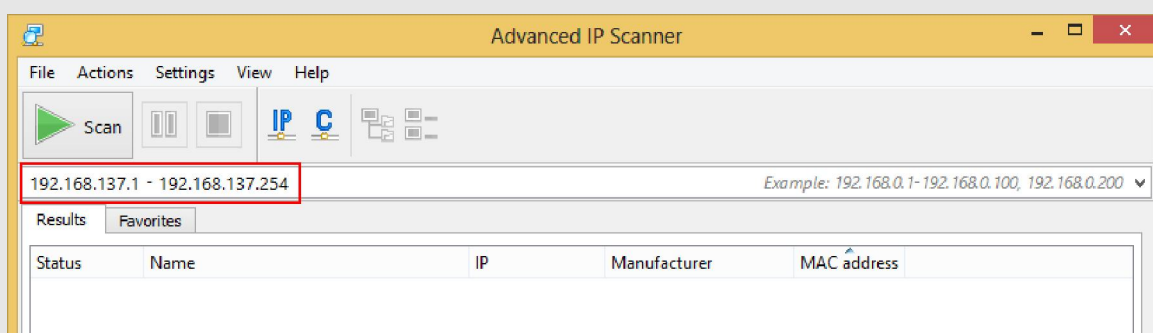
แปลงเป็นเลขฐานสิบจะได้ 192.168.137.0

จึงได้ Network IP มีค่าเท่ากับ 192.168.137.0 หรือเรียกอีกอย่างคือ ค่าเริ่มต้นของ IP แอดเดรสของวง LAN นี้

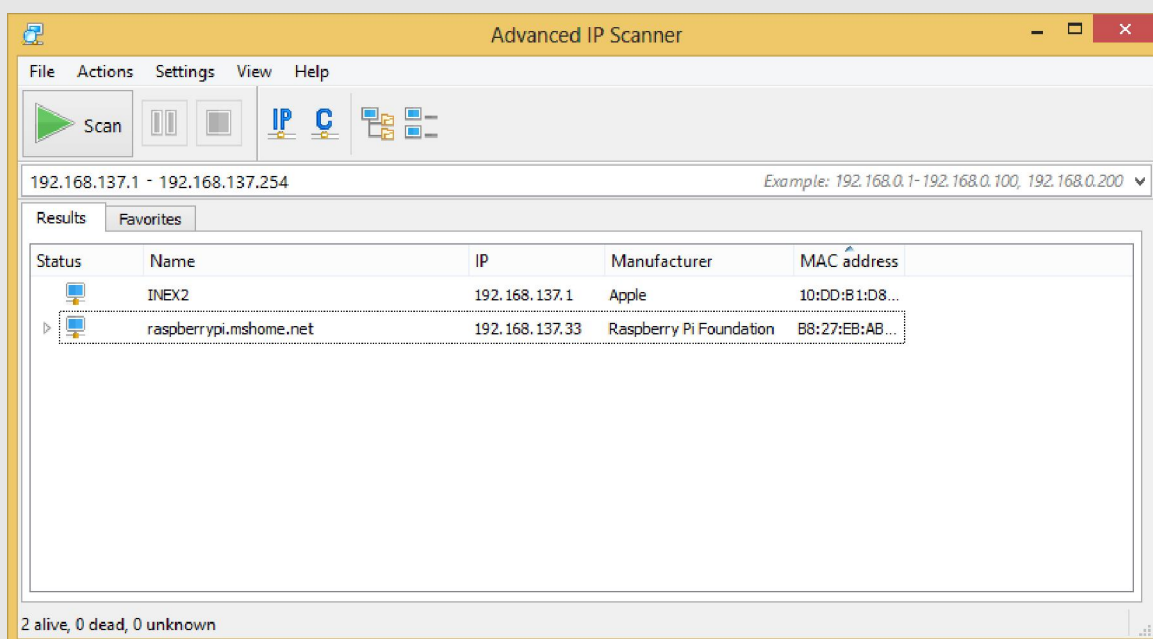
## (6.2) คำค้นหาหมายเลขสิ้นสุดของ IP แอดเดรส หรือ Broadcast IP

วิธีที่ง่ายที่สุดคือ เมื่อทราบแล้วว่า ในแต่ละชุดข้อมูลมีขนาด 8 บิต คือมีค่า 0 ถึง 255 ให้พิจารณาจาก Subnet Mask ก่อน จะเห็นได้ว่าชุดที่ 4 ของ Subnet Mask เป็นเลข 0 นอกนั้นมีค่าเป็นหมายเลข 255 ทั้ง 3 ชุด ดังนั้นหมายเลข Subnet Mask คือ 255.255.255.0 ถึง 255.255.255.255 มีหมายเลขที่เป็นไปได้คือ 256 จำนวน (0 ถึง 255) ส่งผลให้หมายเลข IP แอดเดรสที่อยู่ในวง LAN นี้เริ่มจาก Network IP นั้นคือ 192.168.137.0 ไปถึง 192.168.137.255

(7) เมื่อทราบขอบเขตที่ต้องการหาหมายเลข IP แอดเดรสแล้ว ให้นำหมายเลขนั้นใส่ในช่องแรกของโปรแกรม Advance IP Scanner ดังรูป จากนั้นคลิกที่ปุ่ม SCAN เพื่อเริ่มการค้นหา



(8) เมื่อค้นหาเสร็จแล้ว จะแสดง IP แอดเดรสของอุปกรณ์ทุกตัวที่ต่ออยู่ในวง LAN เดียวกันดังรูป

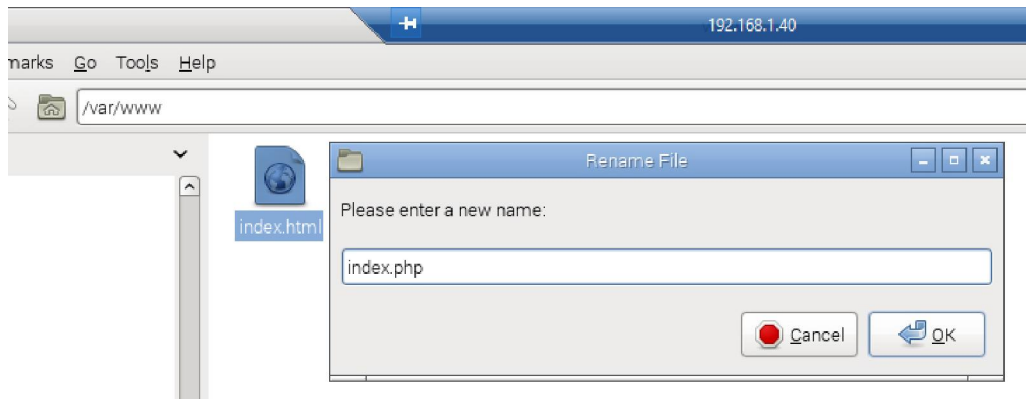


จะได้หมายเลข IP แอดเดรสของบอร์ด Raspberry Pi 2 ที่ต่อภายในเครือข่าย โดยอาจดูจากชื่อหรือชื่อผู้ผลิตก็ได้ จากนั้นนำหมายเลข IP แอดเดรสไปใช้งานต่อไป

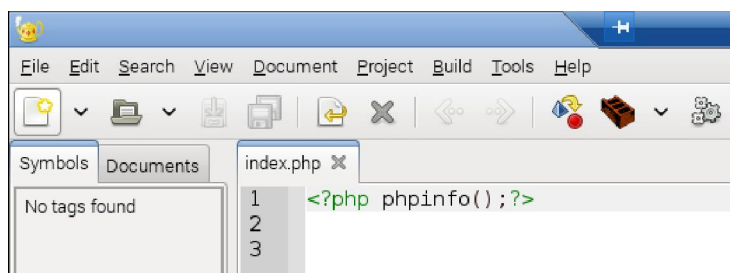


(3) เมื่อติดตั้งเสร็จเรียบร้อยแล้ว ลำดับต่อไปเป็นการทดสอบการทำงาน PHP บน Raspberry Pi

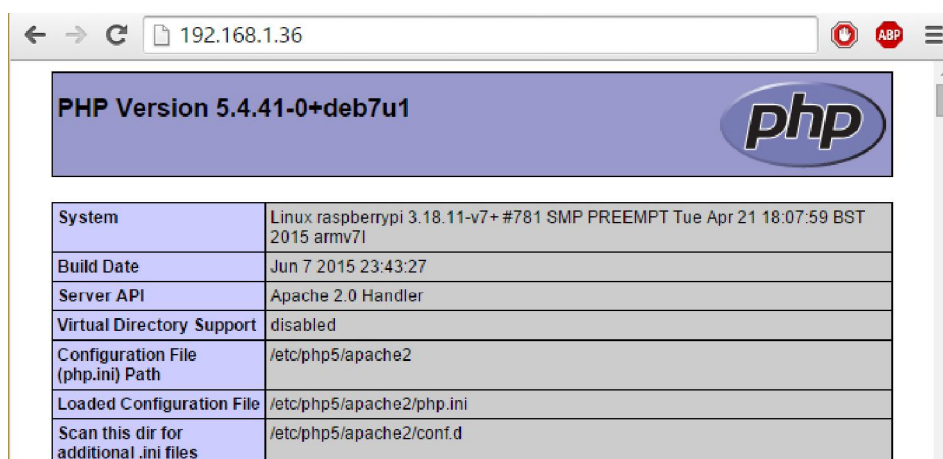
(4) เปิดโฟลเดอร์หรือไดเรกทอรีที่มีชื่อว่า **www** โดยจะอยู่ที่ **/var/www** ภายในมีไฟล์ **index.html** ให้เปลี่ยนนามสกุลเป็น **.php** ดังรูป (ขั้นตอนนี้จะทำได้ก็ต่อเมื่อทำการตั้งค่า Permission จากขั้นตอนที่ผ่านมาเป็นที่เรียบร้อยแล้ว หรือใช้คำสั่งแบบคอมมานด์ไลน์บน Raspberry Pi 2 ศึกษาได้จาก <http://doc.inex.co.th/r-pi-web-server-installation/>)

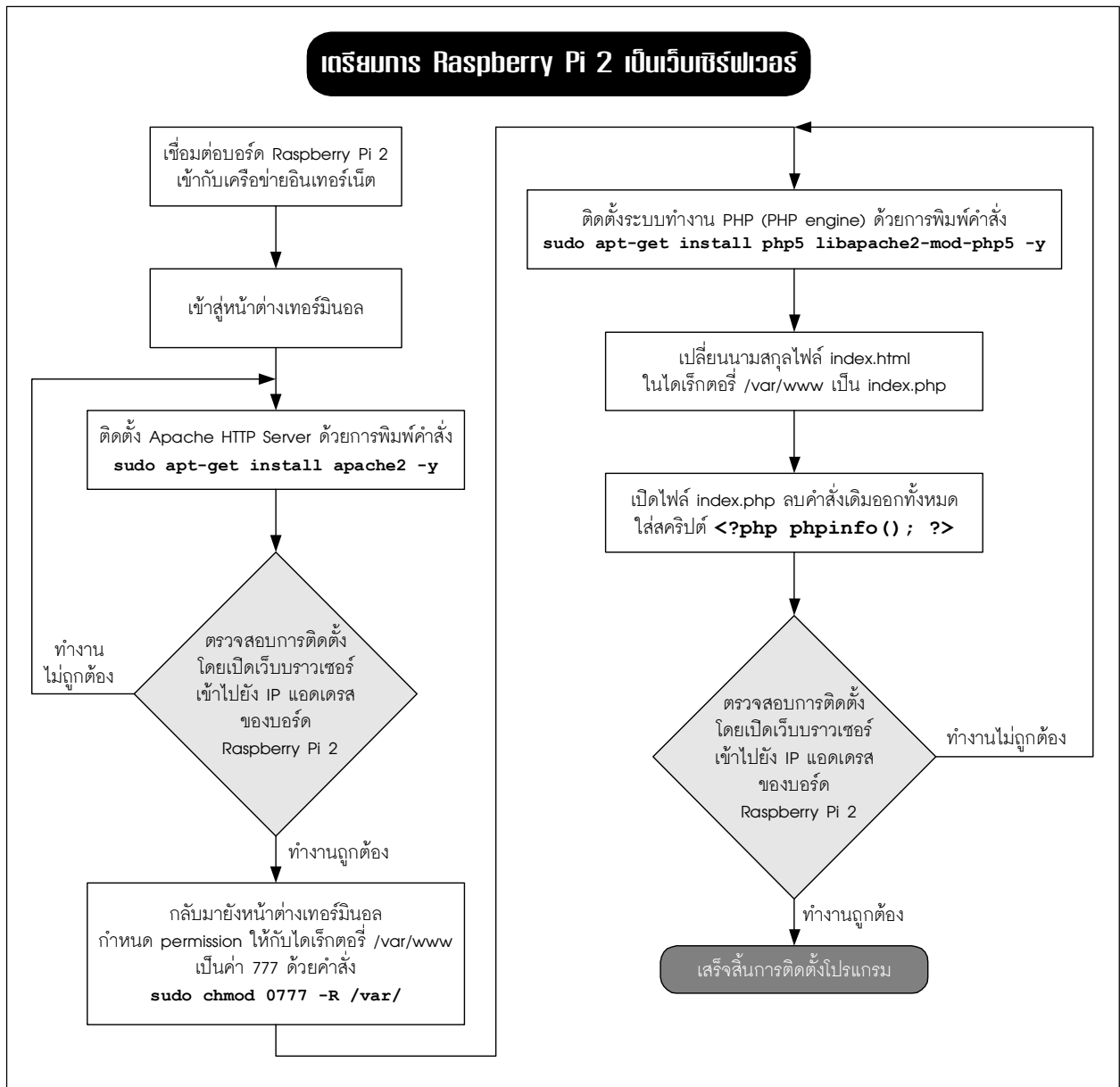


(5) เมื่อเปลี่ยนนามสกุลไฟล์เสร็จแล้ว ลบสคริปต์ที่อยู่ในไฟล์นี้ให้หมด แล้วเพิ่ม `<?php phpinfo(); ?>` ดังรูป จากนั้นบันทึกไฟล์



(6) ทดสอบการทำงานของ PHP โดยเปิดเว็บเบราว์เซอร์ เข้าไปที่ IP แอดเดรสของ Raspberry Pi 2 เช่น **192.168.1.36** จะเห็นเว็บเพจแสดงดังรูป แสดงว่า PHP ทำงานได้ปกติ





รูปที่ 12-4 ผังงานแสดงขั้นตอนการเตรียมการให้ Raspberry Pi 2 ทำงานเป็นเว็บเซิร์ฟเวอร์

## 12.5 รู้จักกับโครงสร้างภาษา HTML

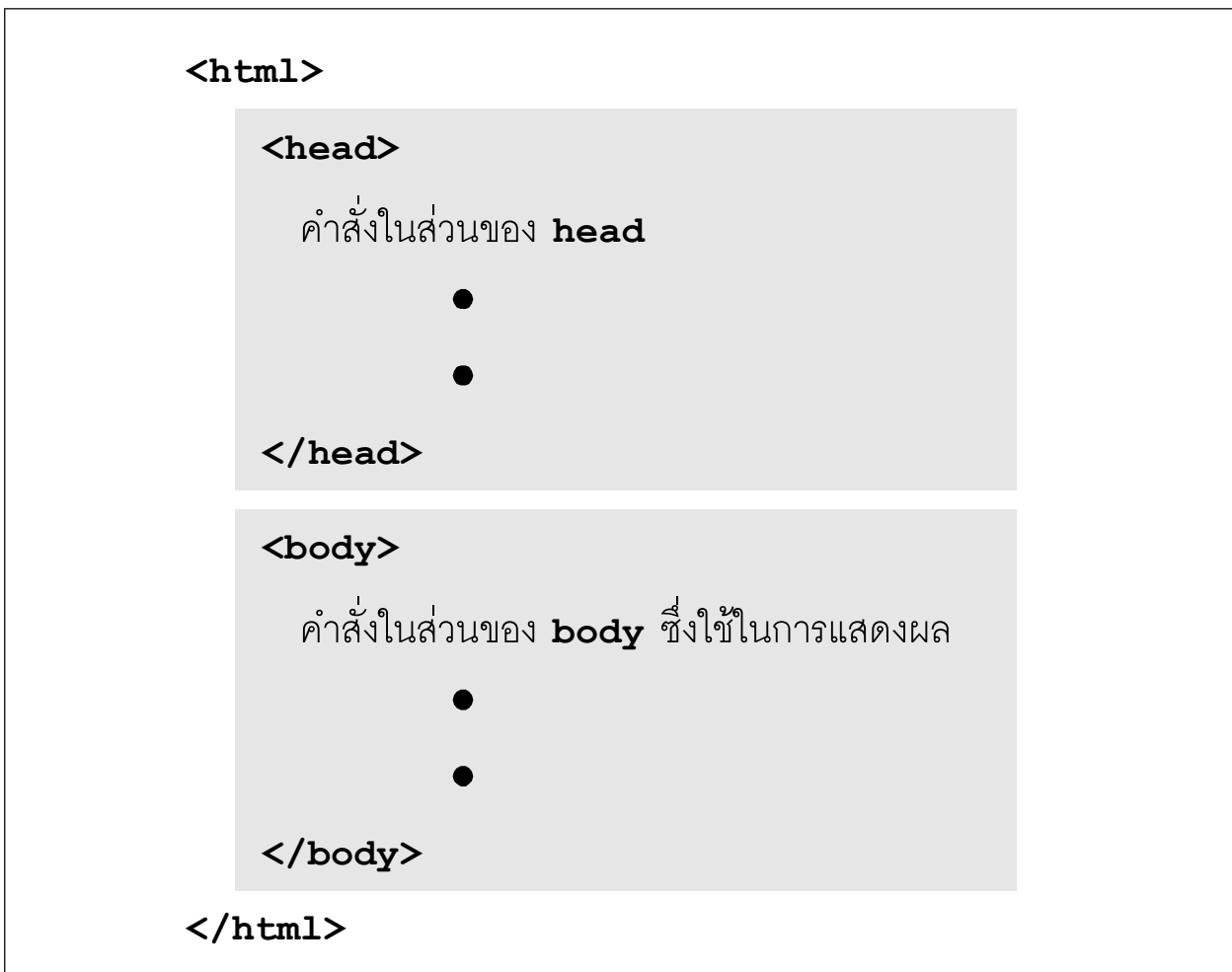
### 12.5.1 ข้อมูลเบื้องต้น

โครงสร้างหลักของ HTML เริ่มต้นด้วย `<html>` และจบด้วย `</html>` เสมอ ดังรูปที่ 12-5 ชุดคำสั่งที่ใช้แยกเป็น 2 ส่วนคือ

1. **head** คำสั่งที่อยู่ในส่วนนี้ใช้บรรยายรายละเอียดเกี่ยวกับเว็บเพจหรือหน้าเว็บ

2. **body** คำสั่งที่อยู่ในส่วนนี้ใช้ในการจัดรูปแบบตัวอักษร จัดหน้า ใส่รูปภาพ ซึ่งตัวอักษรในส่วนนี้จะแสดงที่เว็บเบราว์เซอร์โดยตรง

ส่วนตัวอย่างของโปรแกรมภาษา HTML เพื่อแสดงเว็บเพจแสดงไว้ในโปรแกรมที่ 12-1



รูปที่ 12-5 โครงสร้างของโปรแกรมภาษา HTML

```

<html>
  <head>
    <title>Homepage Title</title>
  </head>
  <body>
    Value: <input type="text" name="fname">
    <button type="button">Click Me!</button><br>
    <input type="submit" value="Submit">
    <table style="width:50%">
      <tr>
        <td>Number</td>
        <td>Value1</td>
        <td>Value2</td>
      </tr>
      <tr>
        <td>1</td>
        <td>data1</td>
        <td>data2</td>
      </tr>
      <tr>
        <td>2</td>
        <td>data3</td>
        <td>data4</td>
      </tr>
    </table>
  </body>
</html>

```

### คำอธิบายชุดคำสั่ง

<title>Homepage Title</title> คำสั่งที่ใช้ระบุชื่อ web page

<input type="text" name="fname"> คำสั่งที่ใช้แสดง Textbox เพื่อรับข้อมูลจาก คีย์บอร์ด

<button type="button">Click Me!</button> คำสั่งที่ใช้แสดงปุ่ม

<br> คำสั่งที่ใช้ขึ้นบรรทัดใหม่

<input type="submit" value="Submit"> คำสั่งแสดงปุ่ม

<table style="width:50%">.....</table> คำสั่งแสดงตารางที่มีความกว้าง 50% ของหน้าจอ

<tr>....</tr> คำสั่งแสดงข้อมูลใน 1 แถว

<td>....</td> คำสั่งแสดงข้อมูลใน 1 สดมภ์

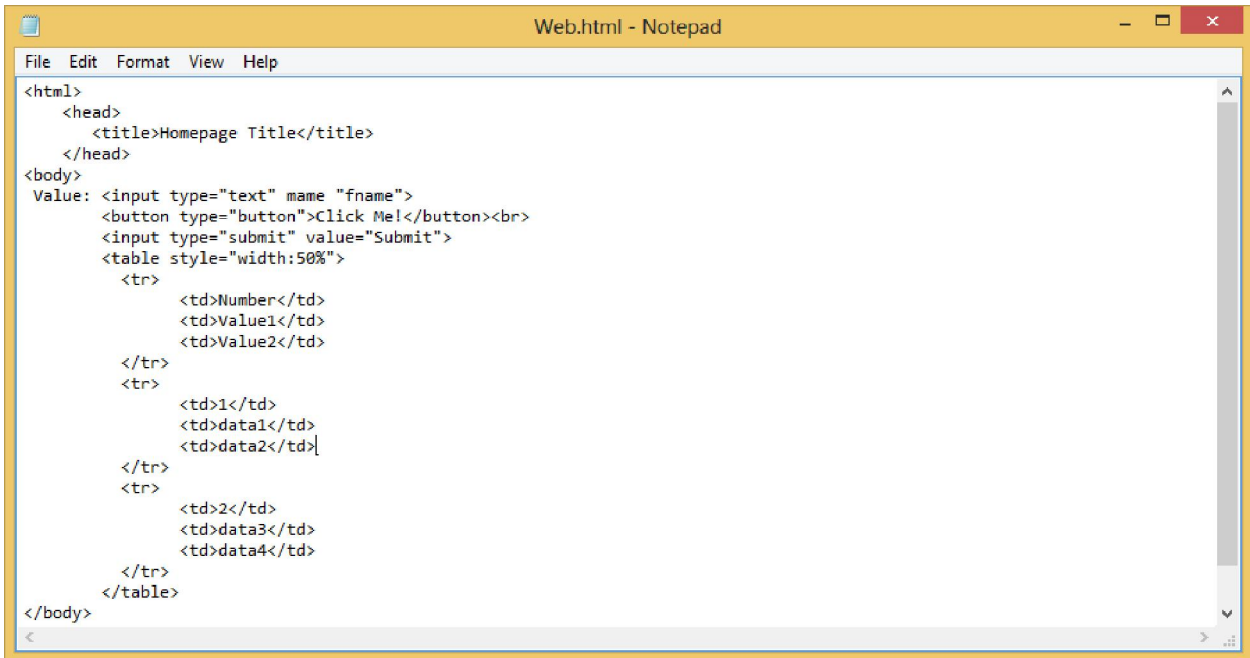
**โปรแกรมที่ 12-1 ตัวอย่างโปรแกรมของโฮมเพจ ที่เขียนขึ้นด้วยภาษา HTML**



## 12.5.2 การทดสอบโปรแกรมภาษา HTML บนคอมพิวเตอร์

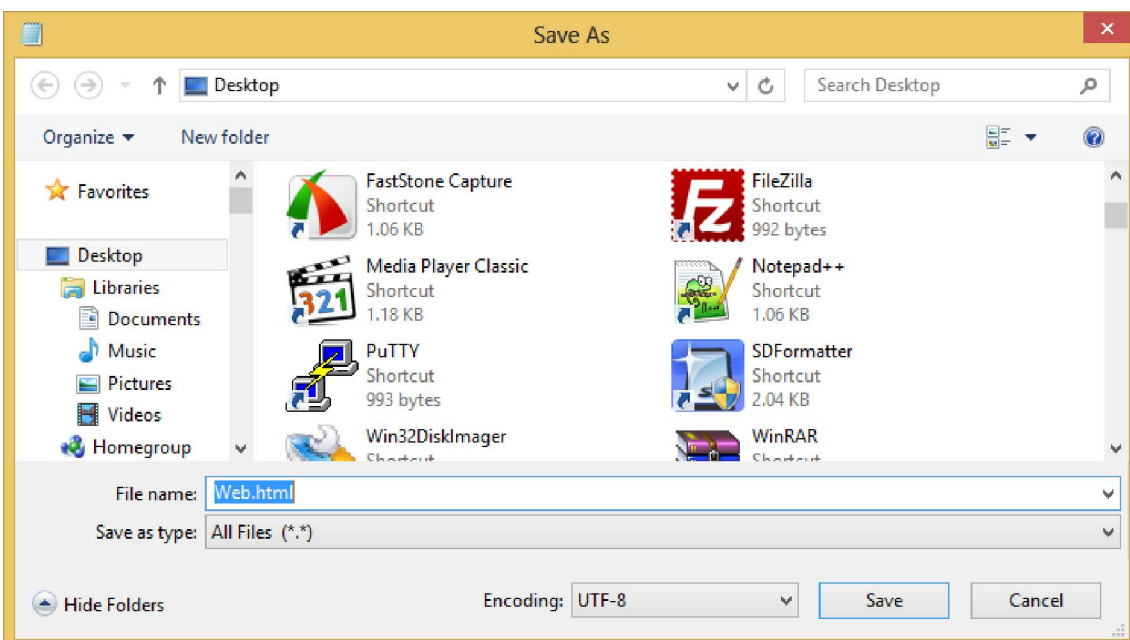
เพื่อให้เกิดความเข้าใจในโปรแกรมภาษา HTML มากขึ้น ให้ทดสอบตามขั้นตอนต่อไปนี้

(1) เปิดโปรแกรม Notepad จากนั้นพิมพ์คำสั่ง HTML จากโปรแกรมที่ 12-1 ดังรูปที่ 12-6



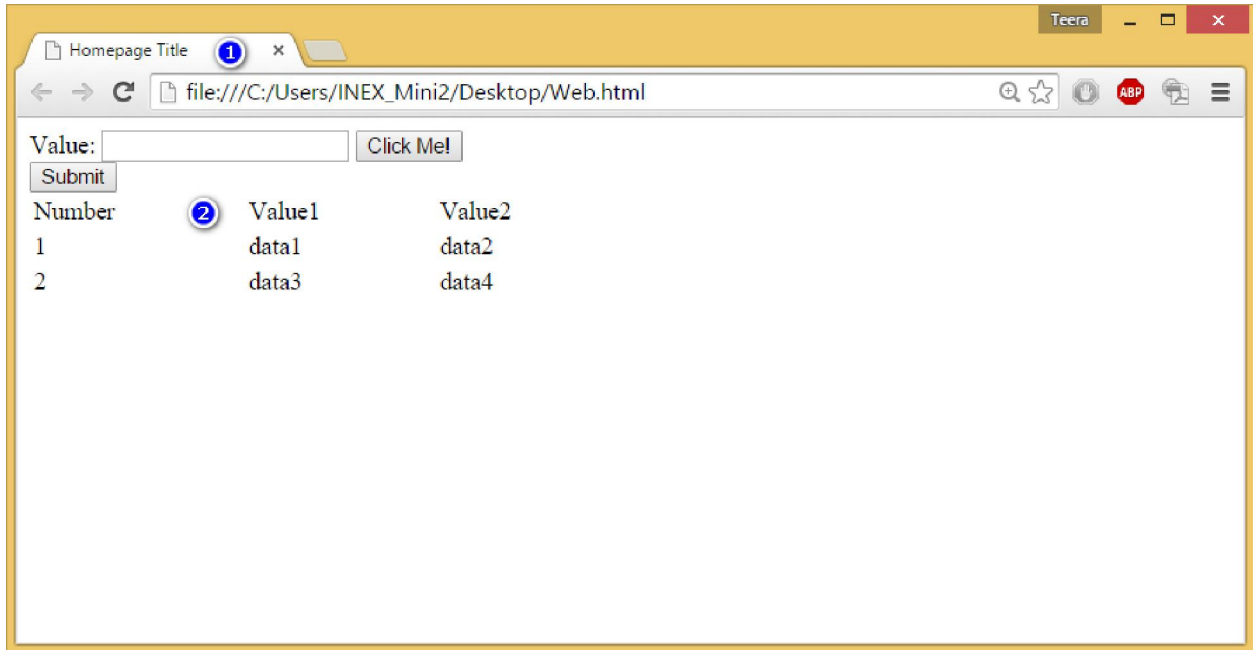
รูปที่ 12-6 ตัวอย่างโปรแกรมภาษา HTML ที่พิมพ์ลงในโปรแกรม Notepad

(2) บันทึกไฟล์ชื่อ **Web.html** เลือกบันทึกแบบ **All File (\*.\*)** และเข้ารหัสแบบ **UTF-8** ดังรูปที่ 12-7 แล้วกดปุ่ม **Save**



รูปที่ 12-7 บันทึกไฟล์ภาษา HTML

(3) เปิดไฟล์ Web.html ด้วยเว็บเบราว์เซอร์ อาทิ Google Chrome หรือ Mozilla Firefox จะได้ผลดังรูปที่ 12-8



รูปที่ 12-8 ตัวอย่างเว็บเพจที่เขียนขึ้นจากโปรแกรมที่ 12-1

หมายเลข 1 เป็นผลของส่วนคำสั่งที่เป็น head

หมายเลข 2 เป็นผลของส่วนคำสั่งที่เป็น body

## 12.6 การสร้างเว็บเพจบนบอร์ด Raspberry Pi 2

จากหัวข้อ 12.5 เป็นการแนะนำให้ทดลองสร้างไฟล์ HTML บนคอมพิวเตอร์ทั่วไป ในหัวข้อนี้เปลี่ยนมาเป็นการสร้างไฟล์ HTML บนบอร์ด Raspberry Pi 2 บ้าง โดยตั้งชื่อไฟล์ HTML ว่า Web.html มีขั้นตอนดังนี้

- (1) เปิดโปรแกรม Geany จากนั้นเลือกสร้างไฟล์ใหม่ โดยคลิกเลือก **File>New(with\_Template)**

> file.html

- (2) เมื่อสร้างไฟล์ขึ้นมาแล้ว เพิ่มชุดคำสั่งเข้าไป ตามรูปที่ 12-9

```

23
24 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
25 "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
26 <html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
27
28   ①
29   <head>
30     <title>Homepage Title</title>
31
32     <meta http-equiv="content-type" content="text/html; charset=utf-8" />
33     <meta name="generator" content="Geany 1.22" />
34   </head>
35
36   ②
37   <body>
38     Value: <input type="text" name="fname">
39     <button type="button">Click Me!</button><br>
40     <input type="submit" value="Submit">
41     <table style="width:50%">
42
43
44
45
46
47
48   </body>
49 </html>

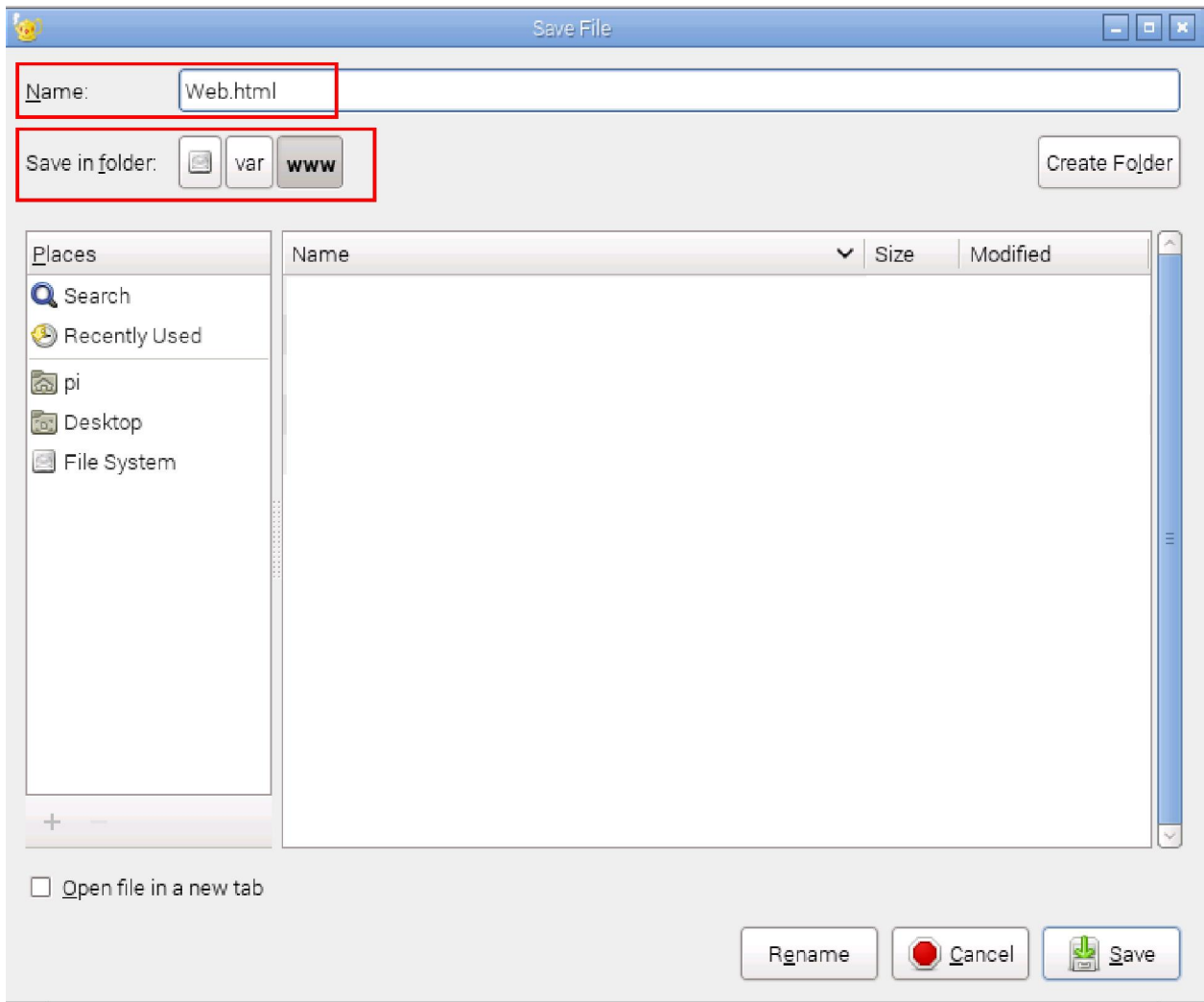
```

รูปที่ 12-9 สร้างไฟล์ HTML บน Geany ที่ติดตั้งบนบอร์ด Raspberry Pi 2

หมายเลข 1 คำสั่งในส่วน head

หมายเลข 2 คำสั่งในส่วน body

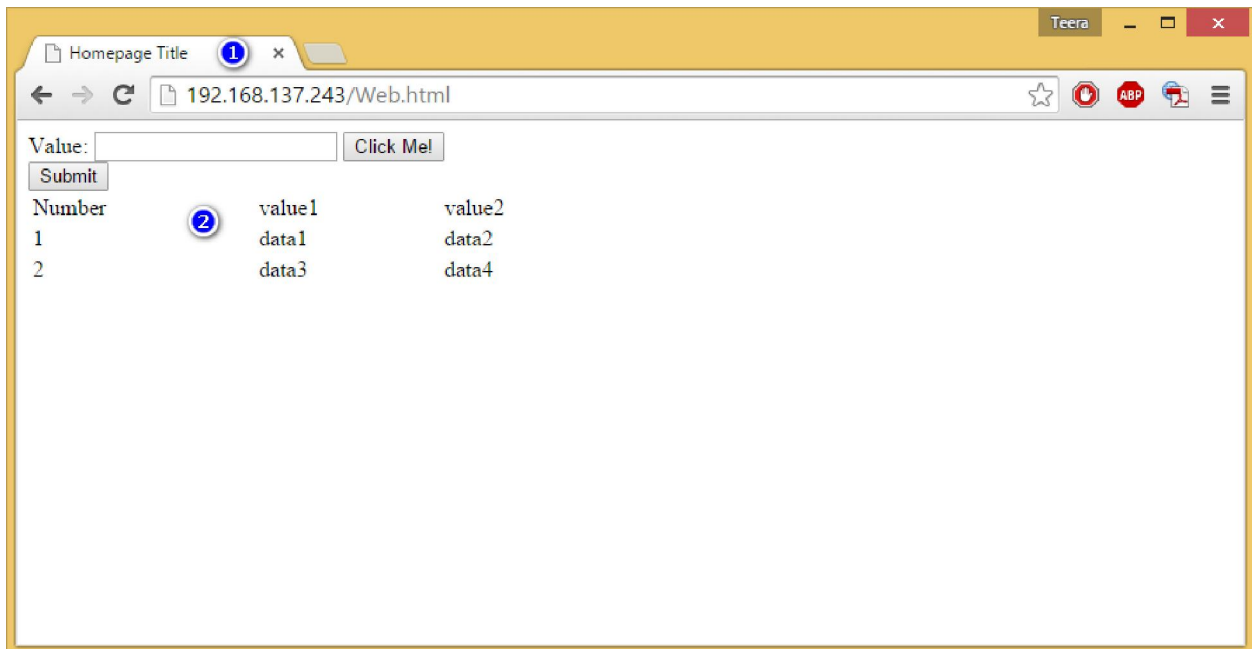
(3) บันทึกไฟล์ที่สร้างขึ้นมา โดยการกด **Ctrl** และ **s** แล้วตั้งชื่อว่า **Web.html** เก็บไว้ที่ **var/www/** ดังรูปที่ 12-10



รูปที่ 12-10 บันทึกไฟล์ HTML บน Geany ที่ติดตั้งบนบอร์ด Raspberry Pi 2

(4) จากนั้นไปที่คอมพิวเตอร์ที่ทำหน้าที่เป็นเว็บไคลเอนต์ ให้พิมพ์ IP แอดเดรสของ Raspberry Pi 2 (ได้มาจากหัวข้อก่อนหน้า) ตามด้วยชื่อไฟล์ที่สร้าง เช่น **192.168.137.243/Web.html** จะได้ผลดังรูปที่ 12-11

**หมายเหตุ :** การหา IP แอดเดรสของ Raspberry Pi 2 อาจใช้โปรแกรมค้นหาหรือใช้คำสั่ง **ifconfig** บน Raspberry Pi 2 เพื่อตรวจสอบก็ได้



รูปที่ 12-11 เว็บเพจที่สร้างขึ้นและเก็บไว้ในบอร์ด Raspberry Pi 2

หมายเลข 1 คำสั่งในส่วน head

หมายเลข 2 คำสั่งในส่วน body

## 12.7 การใช้งาน PHP

PHP คือภาษาคอมพิวเตอร์ในลักษณะเซิร์ฟเวอร์-ไซด์ สคริปต์ (server-side script) ภาษา PHP ใช้สำหรับจัดทำเว็บไซต์ และแสดงผลออกมาในรูปแบบ HTML เป็นส่วนประกอบภายในเว็บเพจ โดยคำสั่งจะปรากฏระหว่าง `<?php...?>` หรือ `<?...?>`

### ตัวอย่างที่ 12-1

```
<?php
    echo "Hello, "."<br>";
    echo "World!";
    echo "<br> "."Hello, ";
    echo "Inex!";
?>
```

### ผลลัพธ์

```
Hello,
World!
Hello, Inex!
```

### อธิบายคำสั่ง

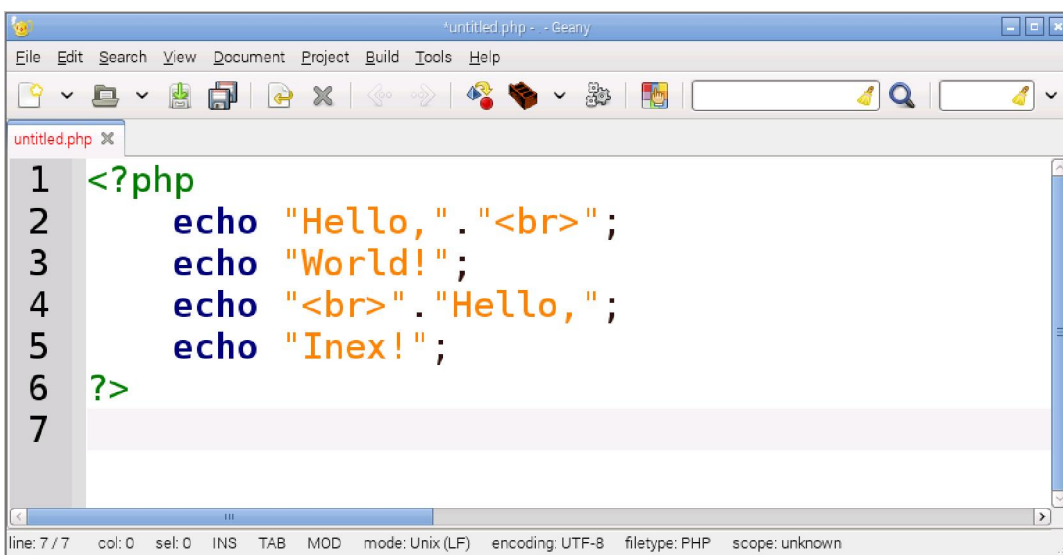
**echo** เป็นคำสั่งที่ใช้แสดงข้อความ การต่อข้อความจะใช้จุด . ส่วนการขึ้นบรรทัดใหม่ใช้รหัส `<br>` ร่วมกับข้อความที่ต้องการขึ้นบรรทัดใหม่ดังตัวอย่างที่ 12-1

## 12.8 สร้างเว็บเพจด้วย PHP

จากการเรียนรู้ในหัวข้อ 12.7 นำข้อมูลและตัวอย่างโปรแกรมต่างๆ มาสร้างเว็บเพจด้วยภาษา PHP บนบอร์ด Raspberry Pi 2 โดยยังคงมีการเชื่อมต่ออุปกรณ์เข้ากับเครือข่ายตามรูปที่ 12-2 หรือ 12-3

(1) สร้างไฟล์ PHP ด้วยการเปิดโปรแกรม Geany เลือกที่ **File > New (with\_Template) > file.php** เมื่อสร้างไฟล์ขึ้นมา

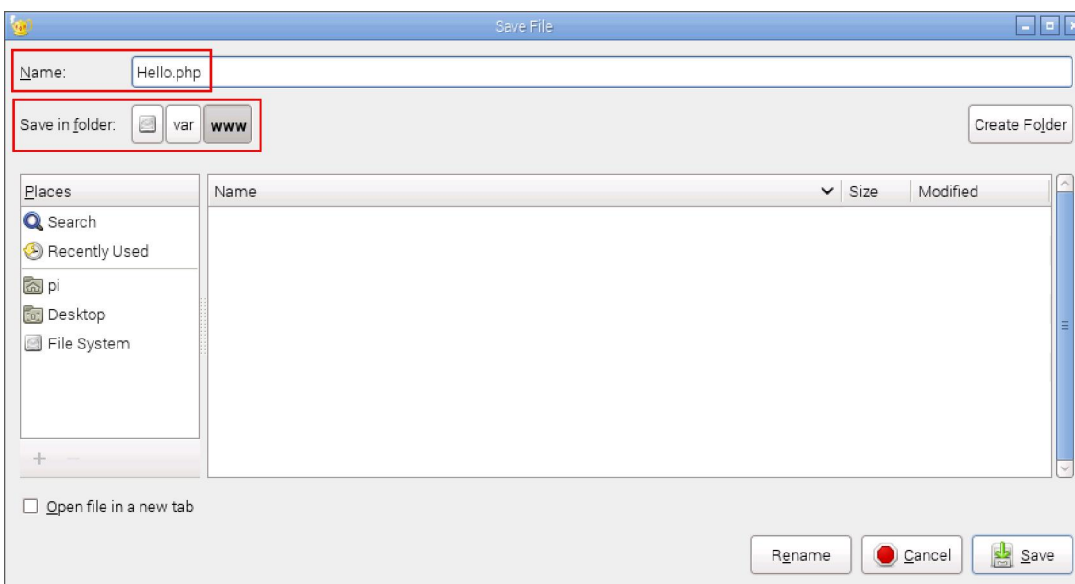
(2) ลบข้อความเดิมออกให้หมด แล้วเพิ่มชุดคำสั่งเข้าไปดังรูปที่ 12-12



```
1 <?php
2     echo "Hello, " . "<br>";
3     echo "World!";
4     echo "<br> " . "Hello, ";
5     echo "Inex!";
6 ?>
7
```

รูปที่ 12-12 หน้าต่างของโปรแกรม Geany ที่แสดงโค้ดสำหรับสร้างไฟล์ PHP

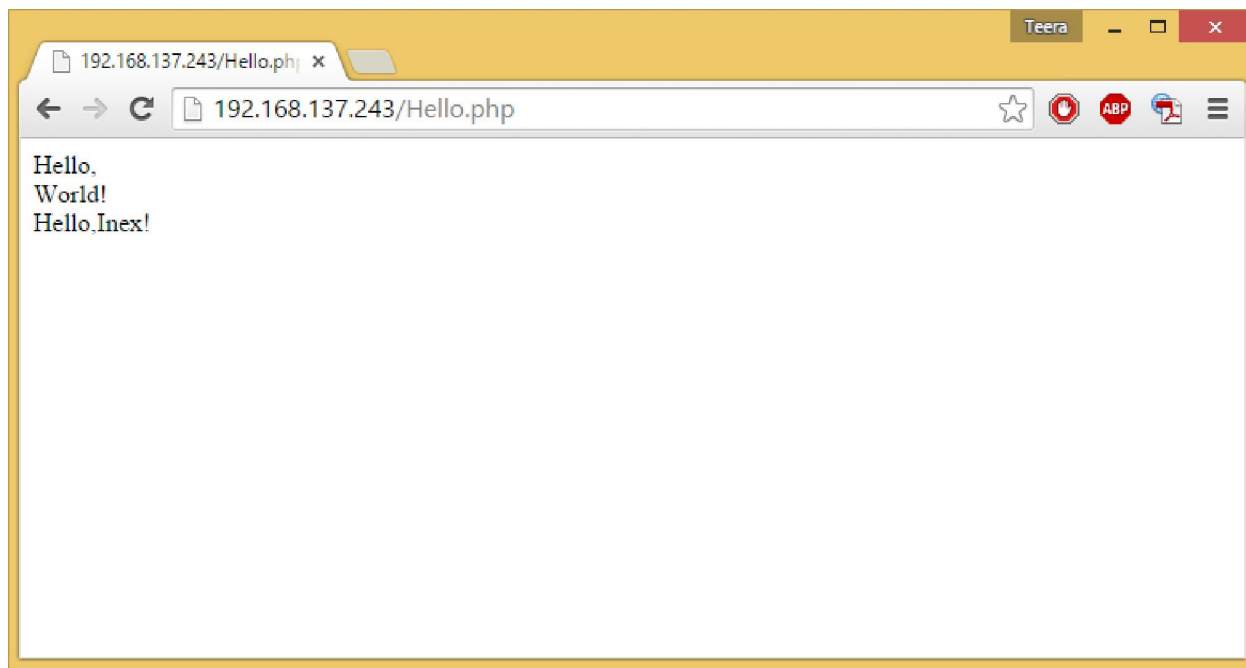
(3) บันทึกไฟล์ โดยการกด **Ctrl** และ **s** ตั้งชื่อเป็น **Hello.php** เก็บไว้ที่ **var/www/** ดังรูปที่ 12-13



รูปที่ 12-13 บันทึกไฟล์ PHP เพื่อสร้างเว็บเพจบนบอร์ด Raspberry Pi 2



(4) จากนั้นไปที่คอมพิวเตอร์ที่ทำหน้าที่เป็นเว็บไคลเอ็นต์ให้พิมพ์ IP แอดเดรสของ Raspberry Pi 2 (ได้มาจากหัวข้อก่อนหน้านี้) ตามด้วยชื่อไฟล์ที่สร้าง เช่น 192.168.137.243/Hello.php จะได้ผลลัพธ์ดังรูปที่ 12-14



รูปที่ 12-14 เว็บเพจที่สร้างขึ้นบนบอร์ด Raspberry Pi 2 ด้วยไฟล์ PHP

## 12.9 คำสั่งพื้นฐานในการควบคุมขาพอร์ต GPIO แบบคอมมานด์ไลน์

ในหัวข้อนี้แนะนำการเขียนโปรแกรมสั้นๆ แบบคอมมานด์ไลน์ (คำสั่งเรียงบรรทัด) เพื่อติดต่อและควบคุมขา GPIO ของ Raspberry Pi 2 เพื่อนำไปใช้เมื่อต้องการสั่งงานขาพอร์ต GPIO จากเว็บไคลเอ็นต์ผ่านเซิร์ฟเวอร์

### รูปแบบคำสั่ง

```
gpio -g mode "pin" "mode"
gpio -g write "pin" "state"
gpio -g read "pin"
```

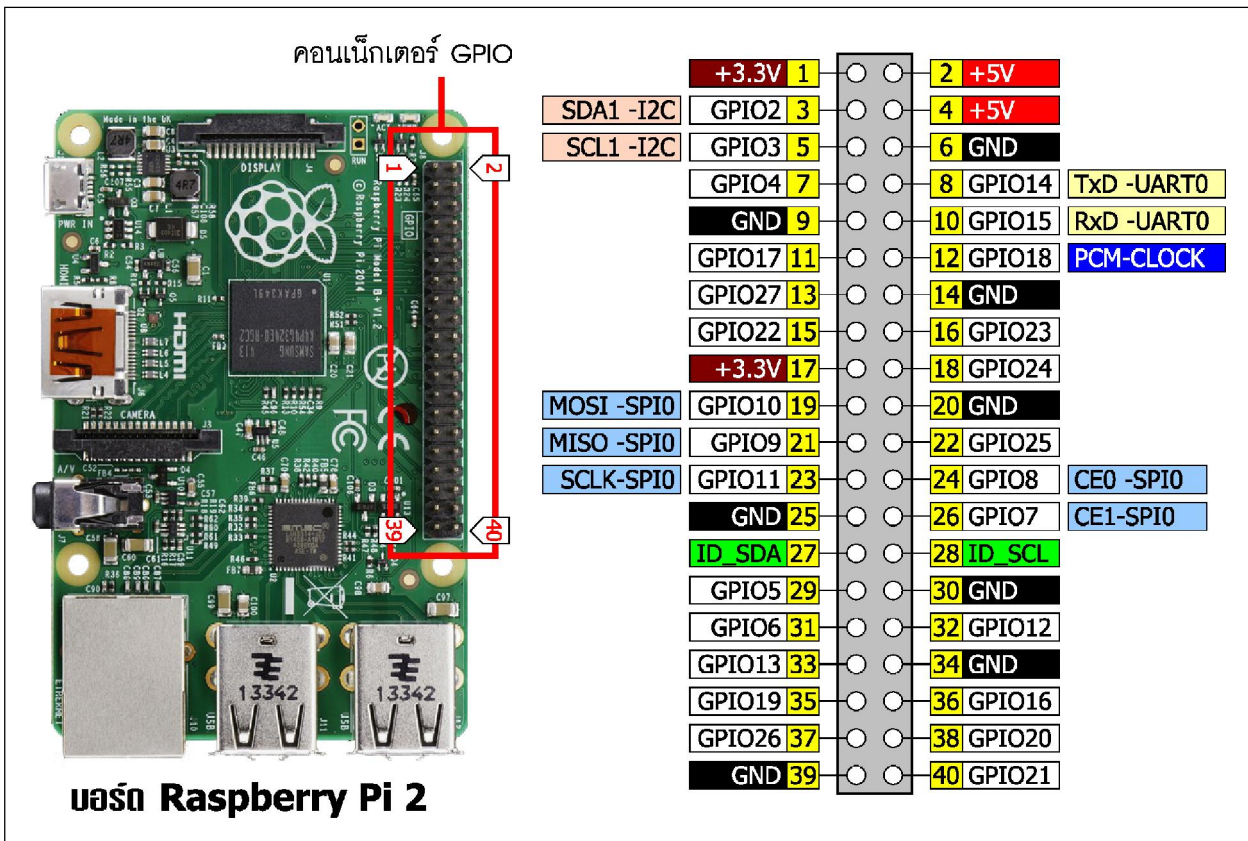
โดยที่

**-g** คือ ใช้ระบุหมายเลข GPIO ตามแบบ BCM\_GPIO ที่ใช้เหมือนกันกับการเขียน Python

**pin** คือ ตำแหน่งขาบน Raspberry Pi

**mode** คือ กำหนดลักษณะการทำงานของขา input, output

**state** คือ บอกสถานะการทำงานของขามี 1 และ 0



รูปที่ 12-15 การจัดขาพอร์ต GPIO ของบอร์ด Raspberry Pi 2

ตัวอย่างที่ 12-2

- `gpio -g mode 16 output` : กำหนดให้ขาพอร์ต 16 เป็นเอาต์พุต
- `gpio -g write 16 1` : กำหนดสถานะลอจิกของขาพอร์ต 16 เป็น "1"
- `gpio -g read 16` : อ่านค่าจากขาพอร์ต 16
- `gpio -g write 16 0` : กำหนดสถานะลอจิกของขาพอร์ต 16 เป็น "0"

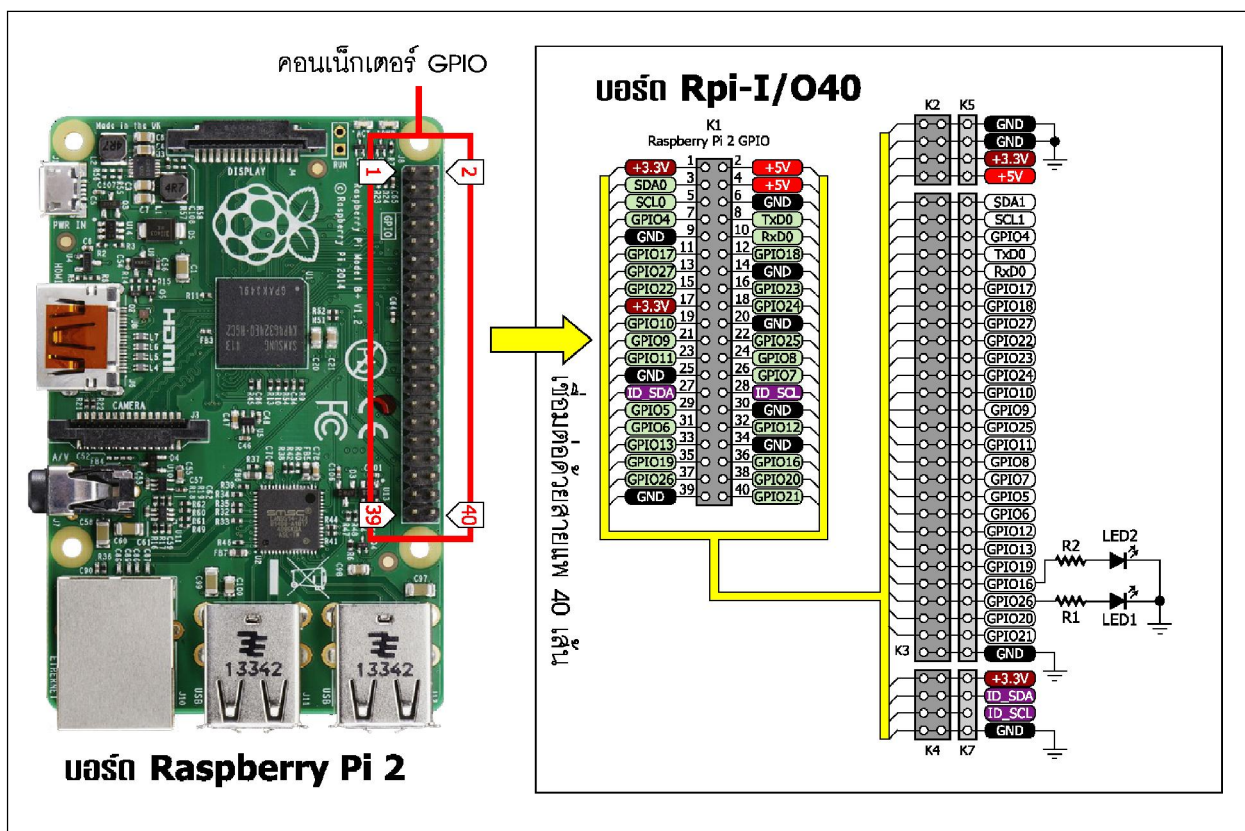
ทำการทดลองตามขั้นตอนต่อไปนี้

- (1) ต้องจรงตามรูปที่ 12-16 กำหนดให้บอร์ด Raspberry Pi 2 เข้าสู่โหมดพร้อมท์
- (2) เขียนโปรแกรมทดสอบดังนี้

```
gpio -g mode 16 output
gpio -g write 16 1
gpio -g read 16
```

จะได้ผลลัพธ์กลับมาเป็น "1" และ LED ที่ต่อกับขาพอร์ต 16 จะติดสว่างตั้งแต่กระทำ

คำสั่ง `gpio -g write 16 1`



รูปที่ 12-16 วงจรสำหรับการทดลองควบคุมขาพอร์ต GPIO ด้วยการเขียนโปรแกรมแบบคอมมานด์ไลน์

(4) เขียนโปรแกรมต่อไปนี้ต่อเนื่องจากข้อ (2)

```
gpio -g write 16 0
```

```
gpio -g read 16
```

จะได้ผลลัพธ์กลับมาเป็น "0" และ LED ที่ต่อกับขาพอร์ต 16 จะดับลง ตั้งแต่กระทำคำสั่ง

```
gpio -g write 16 0
```

ส่วนที่หน้าจอของ Raspberry Pi 2 จะแสดงผลดังรูปที่ 12-17

จะใช้รูปแบบคำสั่งในลักษณะนี้ในการควบคุมการทำงานของขาพอร์ต GPIO บน Raspberry Pi ผ่านเว็บเบราว์เซอร์โดยใช้คำสั่ง system และ exec บนภาษา PHP ในลำดับต่อไป

```
pi@raspberrypi ~ $ gpio -g mode 16 output
pi@raspberrypi ~ $ gpio -g write 16 1
pi@raspberrypi ~ $ gpio -g read 16
1
pi@raspberrypi ~ $ gpio -g write 16 0
pi@raspberrypi ~ $ gpio -g read 16
0
```

รูปที่ 12-17 แสดงการทำงานของบอร์ด Raspberry Pi 2 เมื่อเขียนโปรแกรมแบบคอมมานด์ไลน์เพื่อควบคุมการทำงานของขาพอร์ต GPIO

## 12.10 การรับส่งข้อมูลระหว่างเว็บเพจ

การรับส่งข้อมูลระหว่างเว็บเพจจะใช้เมธอด (method) Get, Post, Put และ Delete สมมติว่ามีแบบฟอร์มให้ผู้ใช้งานป้อนค่า จากนั้นส่งมายังเว็บเซิร์ฟเวอร์ แล้วใช้สคริปต์ PHP เป็นตัวจัดการเก็บข้อมูลลงในฐานข้อมูลหรือเป็นไฟล์สำหรับดูภายหลัง เมธอดที่นิยมใช้จะมี 2 ตัวคือ Get และ Post

### 12.10.1 รูปแบบการใช้เมธอด Get

การใช้เมธอด Get ในการรับข้อมูลจะเห็นข้อมูลที่ส่ง โดยทำการทดสอบได้ง่ายๆ

#### ตัวอย่างที่ 12-3

```
<?php
    if(isset($_GET["f_name"])&isset($_GET["l_name"])){
        echo "first name=".$_GET["f_name"];
        echo "<br>". "last name=".$_GET["l_name"];
    }
?>
```

#### อธิบายคำสั่ง

`$_GET["var name"]` เป็นตัวแปรชนิดหนึ่งที่ใช้อ่านค่าเมื่อใช้เมธอด Get

`isset(var name)` เป็นคำสั่งตรวจสอบว่า ตัวแปรนั้นมีค่าหรือไม่

#### หลักการทำงาน

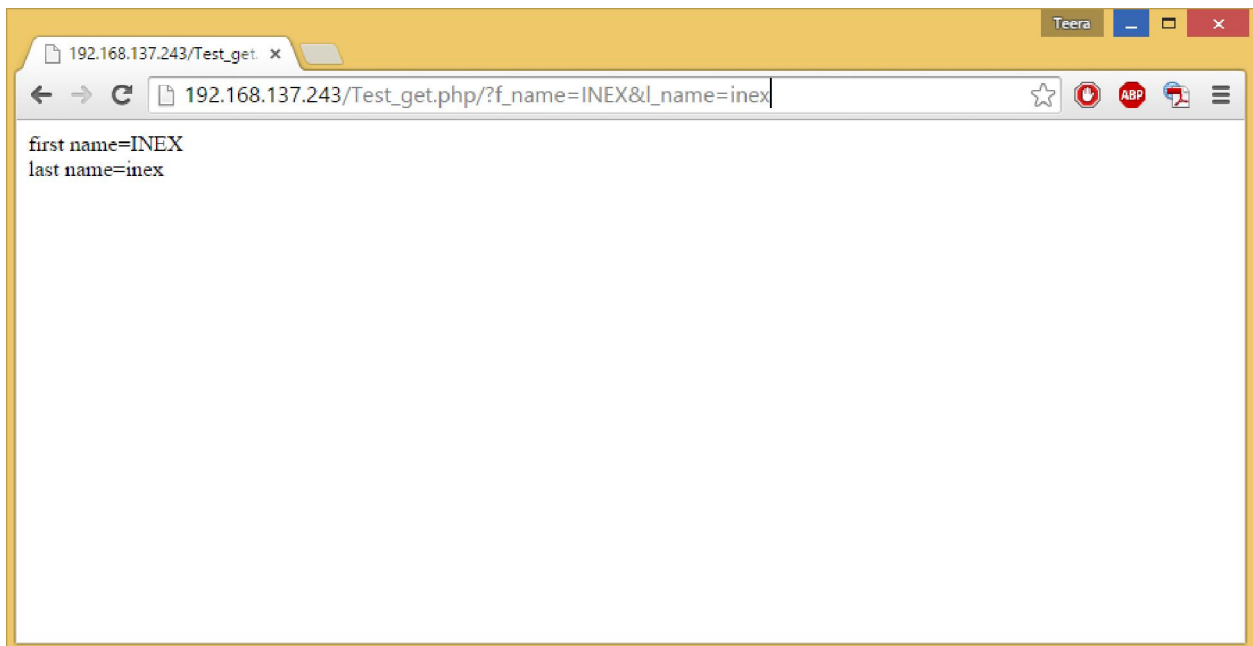
ตรวจสอบว่ามีการส่งข้อมูลทั้งสองตัวมาหรือไม่ ถ้ามีให้แสดงข้อความที่ส่งเข้ามา

#### การทดสอบบน Raspberry Pi 2

(1) ยังคงเชื่อมต่อบอร์ด Raspberry Pi 2 และอุปกรณ์ต่างๆ เข้ากับเครือข่ายตามรูปที่ 12-2 หรือ 12-3

(2) สร้างไฟล์ PHP โดยพิมพ์คำสั่งดังตัวอย่าง ตั้งชื่อไฟล์เป็น `Test_get.php` เก็บไว้ที่ `var/www/`

(3) จากนั้นไปที่คอมพิวเตอร์ที่ทำหน้าที่เป็นเว็บไคลเอ็นต์ เปิดเว็บเบราว์เซอร์ให้พิมพ์ IP แอดเดรสของ Raspberry Pi 2 (ได้มาจากหัวข้อก่อนหน้านี้) ตามด้วยชื่อไฟล์ที่สร้างและค่าต่างๆ เช่น `http://192.168.137.243/Test_get.php/?f_name=INEX&l_name=inex` ดังรูปที่ 12-18



รูปที่ 12-18 แสดงการทำงานของไฟล์ Test\_get.php บนบอร์ด Raspberry Pi 2 ผ่านทางเว็บเบราว์เซอร์

จากการป้อน `192.168.137.243/Test_get.php/?f_name=INEX&l_name=inex` เป็นการระบุค่าเพื่อใช้เมธอด `Get` ในการรับค่า โดยเริ่มจาก `/?f_name=INEX&l_name=inex` จะเห็นได้ว่า ที่โค้ด `f_name=INEX` นั้น `f_name` เป็นตัวระบุค่า ส่วน `INEX` เป็นค่าที่ส่งเข้าไป ดังนั้นที่โค้ด `$_GET["f_name"]` จึงเป็นการอ่านค่าจาก `f_name` การส่งค่าที่ใช้เมธอด `Get` มากกว่าหนึ่งตัวจะใช้เครื่องหมาย `&` เป็นตัวคั่น

### 12.10.2 การใช้เมธอด `Get` ควบคุมขาพอร์ต GPIO ของบอร์ด Raspberry Pi 2

ในหัวข้อนี้เป็นการนำเสนอตัวอย่างการใช้เมธอด `Get` เพื่อควบคุมการทำงานของขาพอร์ต GPIO ของบอร์ด Raspberry Pi 2 ผ่านเว็บเบราว์เซอร์

(1) ยังคงเชื่อมต่อบอร์ด Raspberry Pi 2 และอุปกรณ์ต่างๆ เข้ากับเครือข่ายตามรูปที่ 12-2 หรือ 12-3

(2) ใช้วงจรในรูปที่ 12-16 ในการทดลอง

(3) ทำการสร้างไฟล์ PHP โดยใช้โค้ดในโปรแกรมที่ 12-2

```
<?
if(isset($_GET["st_GPIO16"]))
{
    system("gpio -g mode 16 out");
    system("gpio -g write 16 ".$_GET["st_GPIO16"]);
    exec("gpio -g read 16",$st_LED1);
    echo("LED1=".$_st_LED1[0]."<br>");
}
if(isset($_GET["st_GPIO26"]))
{
    system("gpio -g mode 26 out");
    system("gpio -g write 26 ".$_GET["st_GPIO26"]);
    exec("gpio -g read 26",$st_LED2);
    echo("LED2=".$_st_LED2[0]);
}
?>
```

### อธิบายคำสั่ง

**system** เป็นคำสั่งที่ใช้ตรวจสอบการทำงานของคำสั่งที่ต้องการให้ดำเนินการ ใช้กับคำสั่งที่ไม่ต้องการผลลัพธ์ ข้อดีคือรู้สถานะว่า คำสั่งที่ต้องการทำงานได้หรือไม่ ถ้าดำเนินการสำเร็จ จะส่งค่าหมายเลข 0 กลับมา

**exec** เป็นคำสั่งสำหรับสั่งการให้คำสั่งที่ต้องการทำงาน ใช้กับคำสั่งที่ต้องการผลลัพธ์มากกว่า 1 ค่า โดยผลลัพธ์จะอยู่ในรูปแบบอะเรย์ ดังนั้นเมื่อใช้งานจะต้องกำหนดขนาดของอะเรย์ด้วย แม้ว่าจะมีผลลัพธ์เพียงค่าเดียวก็ตาม

การทำงานของโปรแกรม มีลำดับดังนี้

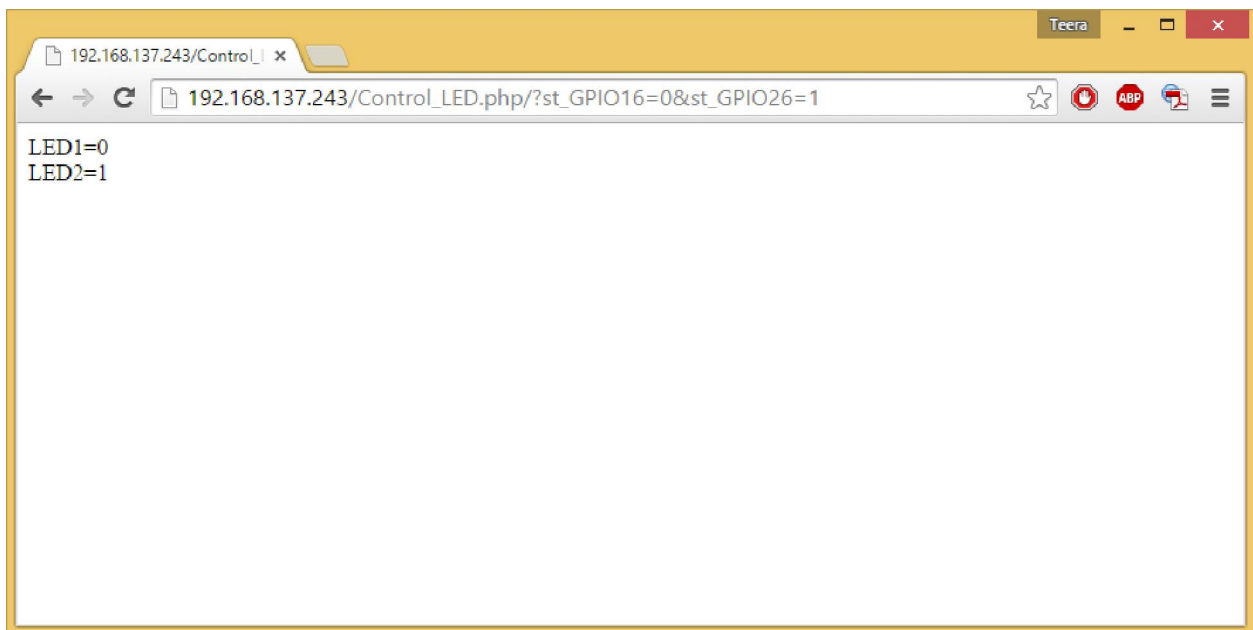
- (1) ตรวจสอบว่ามีข้อมูลส่งมาหรือไม่
- (2) กำหนดการทำงานของขาพอร์ต GPIO16 ให้เป็นเอาต์พุต
- (3) กำหนดสถานะลอจิกของขาพอร์ต GPIO16 ตามข้อมูลที่รับเข้ามา
- (4) อ่านค่าสถานะลอจิกของขาพอร์ต GPIO16 ณ ปัจจุบัน
- (5) แสดงผล

โปรแกรมที่ 12-2 ไฟล์ **Control\_LED.php** ที่ใช้ควบคุมการทำงานของขาพอร์ต GPIO 16 และ 26 ของบอร์ด Raspberry Pi 2 ผ่านเว็บเบราว์เซอร์

(4) บันทึกเป็นไฟล์ชื่อ **Control\_LED.php** เก็บไว้ที่ **var/www/**

(5) จากนั้นไปที่คอมพิวเตอร์ที่ทำหน้าที่เป็นเว็บไคลเอ็นต์ เปิดเว็บเบราว์เซอร์ให้พิมพ์ IP แอดเดรสของ Raspberry Pi 2 (ได้มาจากหัวข้อก่อนหน้านี้) ตามด้วยชื่อไฟล์ที่สร้างและค่าต่างๆ เช่น **192.168.137.243/Control\_LED.php/?st\_GPIO16=0&st\_GPIO26=1** ดังรูปที่ 12-19 สังเกตการทำงานของ LED

(6) ทดลองเปลี่ยนค่าของ **st\_GPIO16** และ **st\_GPIO26** ไม่ให้เหมือนเดิม แล้วสังเกตการทำงานของ LED



รูปที่ 12-19 แสดงการทำงานของไฟล์ **Control\_LED.php** บนบอร์ด Raspberry Pi 2 ผ่านทางเว็บเบราว์เซอร์



## 12.11 การสั่งให้สคริปต์ของโปรแกรมภาษา Python ทำงานด้วย PHP

### 12.11.1 กำหนดสิทธิ์ให้แก่ PHP

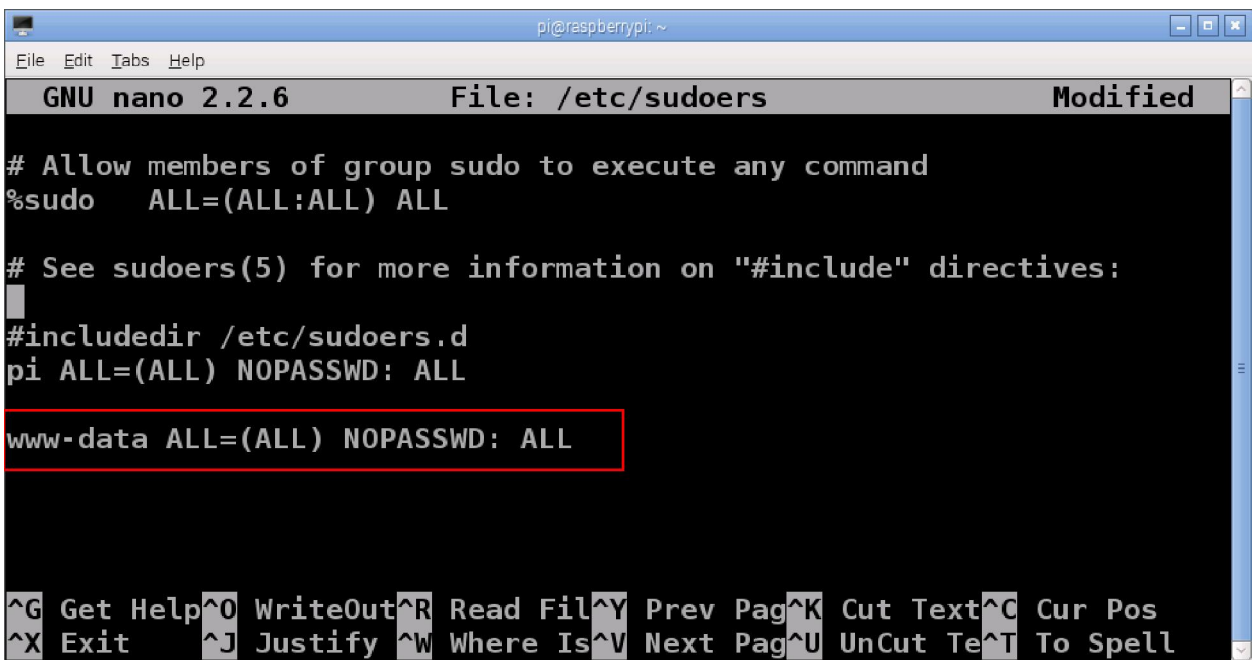
มีขั้นตอนดังนี้

(1) เริ่มด้วยการให้สิทธิ์ PHP เพื่อสั่งให้สคริปต์ของโปรแกรมภาษา Python ทำงานได้ โดยเปิด **LXTerminal** แล้วพิมพ์คำสั่ง `sudo nano /etc/sudoers` เพื่อเข้าไปเพิ่มสิทธิ์ผู้ใช้ ดังรูปที่ 12-20 แล้วกด **Enter**

```
pi@raspberrypi ~ $ sudo nano /etc/sudoers
```

รูปที่ 12-20

(2) พิมพ์ `www-data ALL=(ALL) NOPASSWD: ALL` ไว้ท้ายสุดของไฟล์ ดังรูปที่ 12-21



```
pi@raspberrypi: ~
GNU nano 2.2.6 File: /etc/sudoers Modified
# Allow members of group sudo to execute any command
%sudo  ALL=(ALL:ALL) ALL

# See sudoers(5) for more information on "#include" directives:
#includedir /etc/sudoers.d
pi ALL=(ALL) NOPASSWD: ALL
www-data ALL=(ALL) NOPASSWD: ALL

^G Get Help ^O WriteOut ^R Read Fil ^Y Prev Pag ^K Cut Text ^C Cur Pos
^X Exit ^J Justify ^W Where Is ^V Next Pag ^U UnCut Te ^T To Spell
```

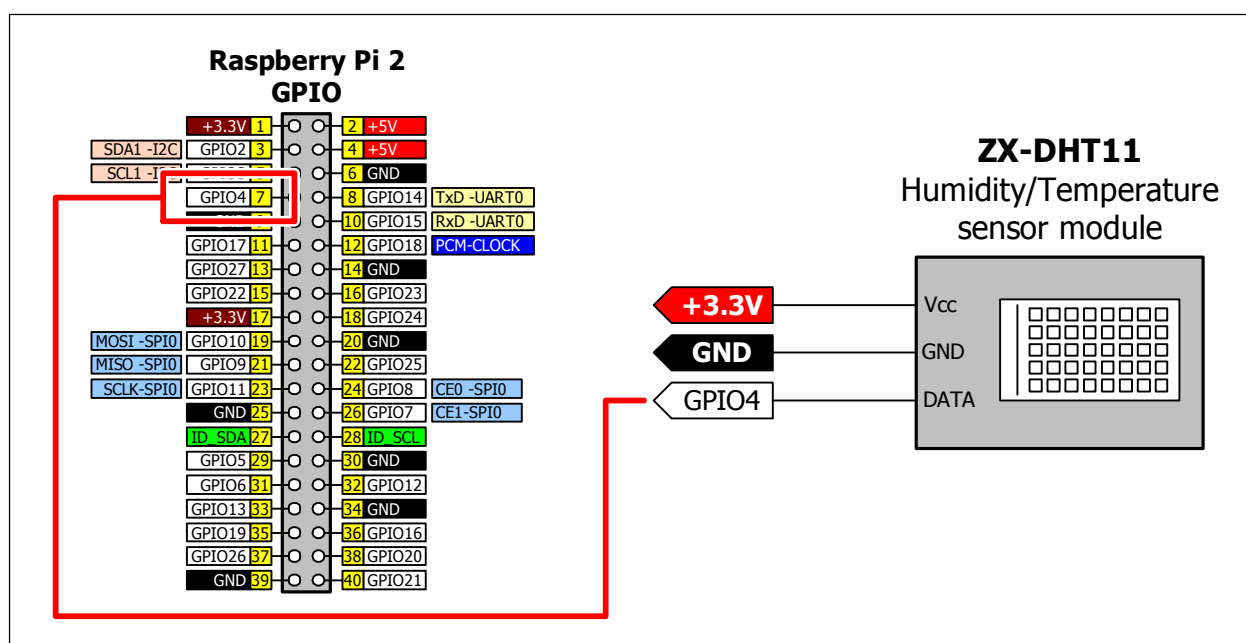
รูปที่ 12-21

(3) จากนั้นให้กด **Ctrl** และ **x** แล้วกดคีย์ **y** เพื่อบันทึกไฟล์ ตามด้วยการกด **Enter** จะเห็นว่า มีการกำหนดให้กลุ่ม `www-data` เป็นกลุ่มผู้ใช้งานในระบบเทียบเท่ากับกลุ่มผู้ใช้งานในระบบที่ชื่อว่า `pi` และ `www-data` เป็นชื่อกลุ่มผู้ใช้งานในส่วน of เว็บด้วย

## 12.11.2 ตัวอย่างการทดสอบรันสคริปต์ Python ด้วย PHP

ในตัวอย่างนี้เป็นการอ่านค่าจากโมดูลวัดความชื้นและอุณหภูมิ DHT11 ด้วย Raspberry Pi 2 โดยสร้างโปรแกรมภาษา Python ที่จะให้ PHP สั่งให้ทำงาน โปรแกรมที่สร้างขึ้นนั้นต้องไม่เป็นโปรแกรมที่ทำงานแบบวนซ้ำ เพราะจะทำให้เว็บเพจต้องรอไปเรื่อยๆ จนกว่าจะหมดเวลา

- (1) ต่อดวงจรรูปที่ 12-22
- (2) เปิดโปรแกรม Geany เพื่อเขียนโปรแกรมตามโปรแกรมที่ 12-3
- (3) บันทึกไฟล์เก็บไว้ที่ `var/www/` โดยตั้งชื่อไฟล์ว่า `Read_DHT11.py`



รูปที่ 12-22 วงจรเชื่อมต่อ DHT11 โมดูลวัดความชื้นสัมพัทธ์และอุณหภูมิของบอร์ด Raspberry Pi 2 ผ่านพอร์ต GPIO เพื่ออ่านค่าด้วยโปรแกรมภาษา Python ซึ่งได้รับการสั่งการมาจาก PHP

```
import time
import Adafruit_DHT
Sensor = Adafruit_DHT.DHT11
GPIO = 4
humidity, temperature = Adafruit_DHT.read_retry(Sensor, GPIO)
if humidity is not None and temperature is not None:
    print(temperature)
    print(humidity)
else:
    print('Failed to get reading. Try again!')
```

### การทำงานของโปรแกรม

อ่านค่าจากโมดูล DHT11 แล้วเก็บไว้ในตัวแปร humidity และ temperature จากนั้นทำการตรวจสอบว่ามีค่าหรือไม่ ถ้ามีให้แสดงผลอุณหภูมิกับความชื้นในอากาศ ถ้าไม่มีให้แสดงข้อความว่า Failed to get reading. Try again!

**โปรแกรมที่ 12-3 ไฟล์ Read\_DHT11.py เป็นโปรแกรมภาษา Python สำหรับอ่านค่าโมดูลวัดความชื้นสัมพัทธ์และอุณหภูมิ DHT11 ด้วยบอร์ด Raspberry Pi 2 ภายใต้การควบคุมจาก PHP**

```
<?
exec("sudo python3 /var/www/Read_DHT11.py", $DHT11);
echo ("Temp=" . $DHT11[0] . "*C") ;
echo ("<br>Humidity=" . $DHT11[1] . "%") ;
?>
```

### อธิบายคำสั่ง

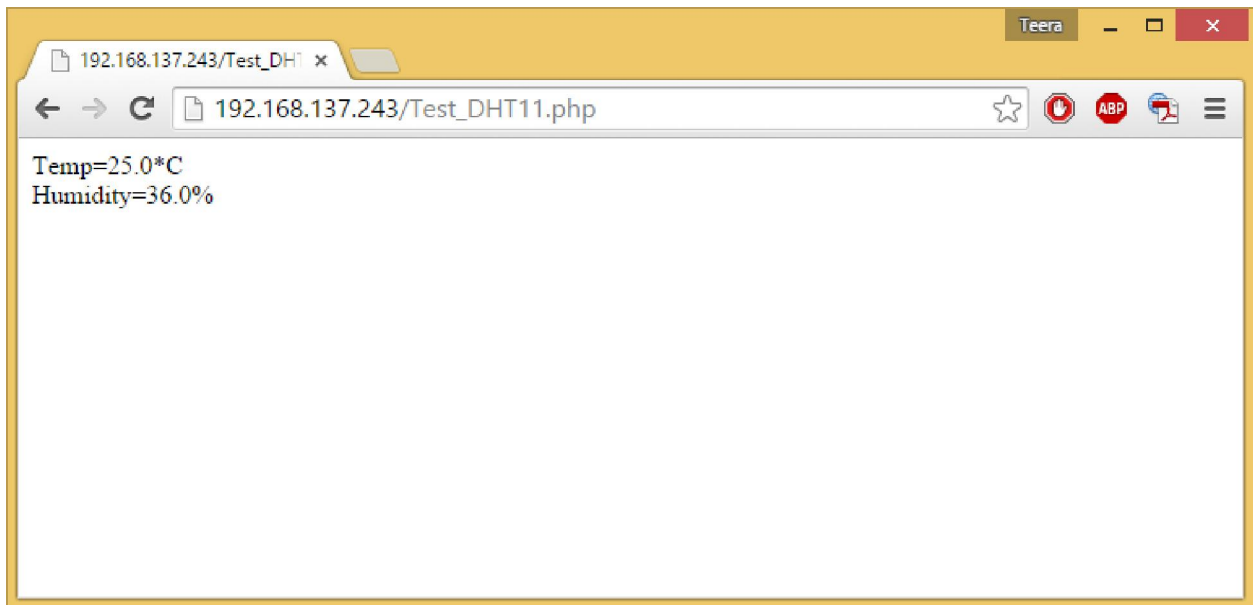
**exec("sudo python3 /var/www/Read\_DHT11.py", \$DHT11);** สั่งให้ไฟล์ที่ชื่อว่า Read\_DHT11.py ที่ถูกเก็บไว้ใน var/www/ ทำงานด้วย Python3 แล้วเก็บผลลัพธ์จากทำงานไว้ในตัวที่ชื่อว่า \$DHT11

**echo ("Temp=" . \$DHT11[0] . "\*C")** แสดงข้อความโดยนำค่าจากตัวแปร \$DHT11 ช่องแรก มาแสดงโดยถูกระบุเป็นค่าอุณหภูมิ จากตัวอย่าง Python เมื่อมีค่าจะทำการแสดงผลอุณหภูมิก่อนแล้วตามด้วยค่าความชื้น ดังนั้นค่าในตัวแปร \$DHT11[0] จึงเป็น ค่าอุณหภูมิและ \$DHT11[1] เป็นค่าความชื้นตามลำดับ

**โปรแกรมที่ 12-4 ไฟล์ Test\_DHT11.php โปรแกรมภาษา PHP เพื่อควบคุมการทำงานของสคริปต์โปรแกรมภาษา Python ไฟล์ Test\_DHT11.py (โปรแกรมที่ 12-3)**

(4) เขียนโปรแกรมภาษา PHP ตามโปรแกรมที่ 12-4 เก็บไว้ที่ `var/www/` แล้วตั้งชื่อว่า **Test\_DHT11.php**

(5) ทำการทดสอบโดยไปที่คอมพิวเตอร์ที่ทำหน้าที่เป็นเว็บไคลเอ็นต์ เปิดเว็บเบราว์เซอร์ ให้พิมพ์ IP แอดเดรสของ Raspberry Pi 2 (ได้มาจากหัวข้อก่อนหน้านี้) ตามด้วยชื่อไฟล์ที่สร้างและค่าต่างๆ เช่น `192.168.137.243/Test_DHT11.php` จะได้ผลการทำงานดังรูปที่ 12-23



รูปที่ 12-23 หน้าต่างเว็บเบราว์เซอร์แสดงเว็บเพจของการแสดงค่าความชื้นสัมพัทธ์และอุณหภูมิที่พัฒนาจากโปรแกรมภาษา PHP โดยอ่านข้อมูลจากบอร์ด Raspberry Pi 2 ซึ่งเขียนโปรแกรมภาษา Python ในการติดต่อกับตัวตรวจจับผ่านทางขาพอร์ต GPIO

## 12.12 การทดลองควบคุมอุปกรณ์เอาต์พุตอย่างง่ายผ่านเว็บเบราว์เซอร์โดยใช้ PHP และ HTML

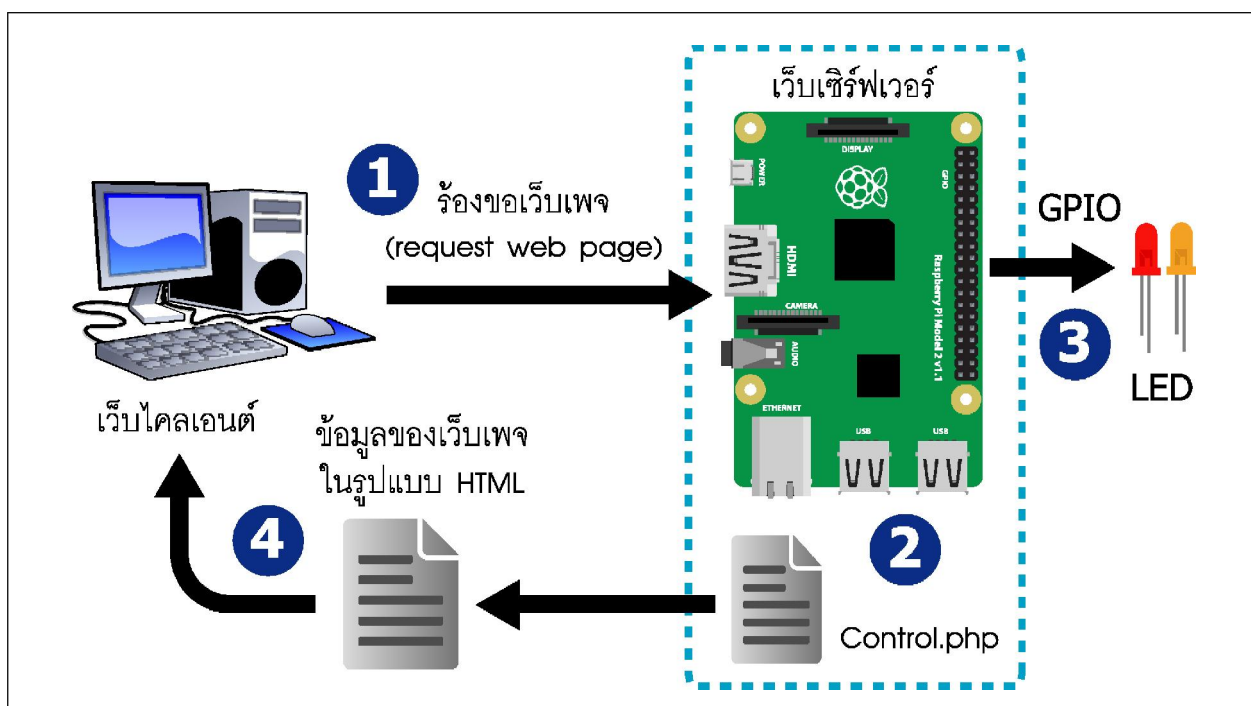
ถึงแม้ว่าด้วยคุณสมบัติของ PHP ที่ช่วยให้การเปลี่ยนข้อความและการตรวจสอบนั้นง่าย ทว่าการสร้างเว็บเพจที่ให้ผู้ใช้งานได้ง่ายนั้น ยังต้องอาศัยการพัฒนาโปรแกรมด้วยภาษา HTML เพื่อให้ผู้พัฒนาสามารถสร้างปุ่มกดหรือเพิ่มช่องกรอกข้อความได้ง่ายขึ้น

ในหัวข้อนี้นำเสนอตัวอย่างการทดลองสร้างสคริปต์ที่พัฒนาด้วยภาษา PHP เพื่อสั่งงานโดยตรงกับขาพอร์ต GPIO ของบอร์ด Raspberry Pi 2

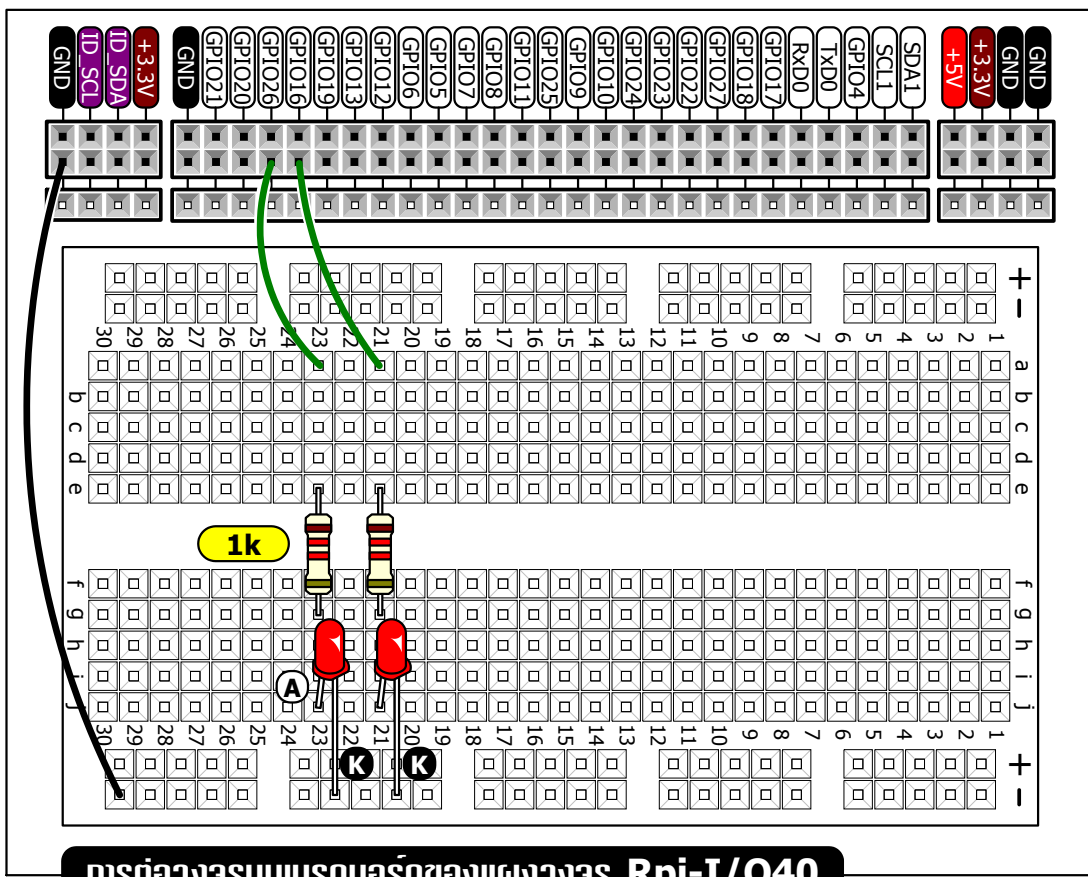
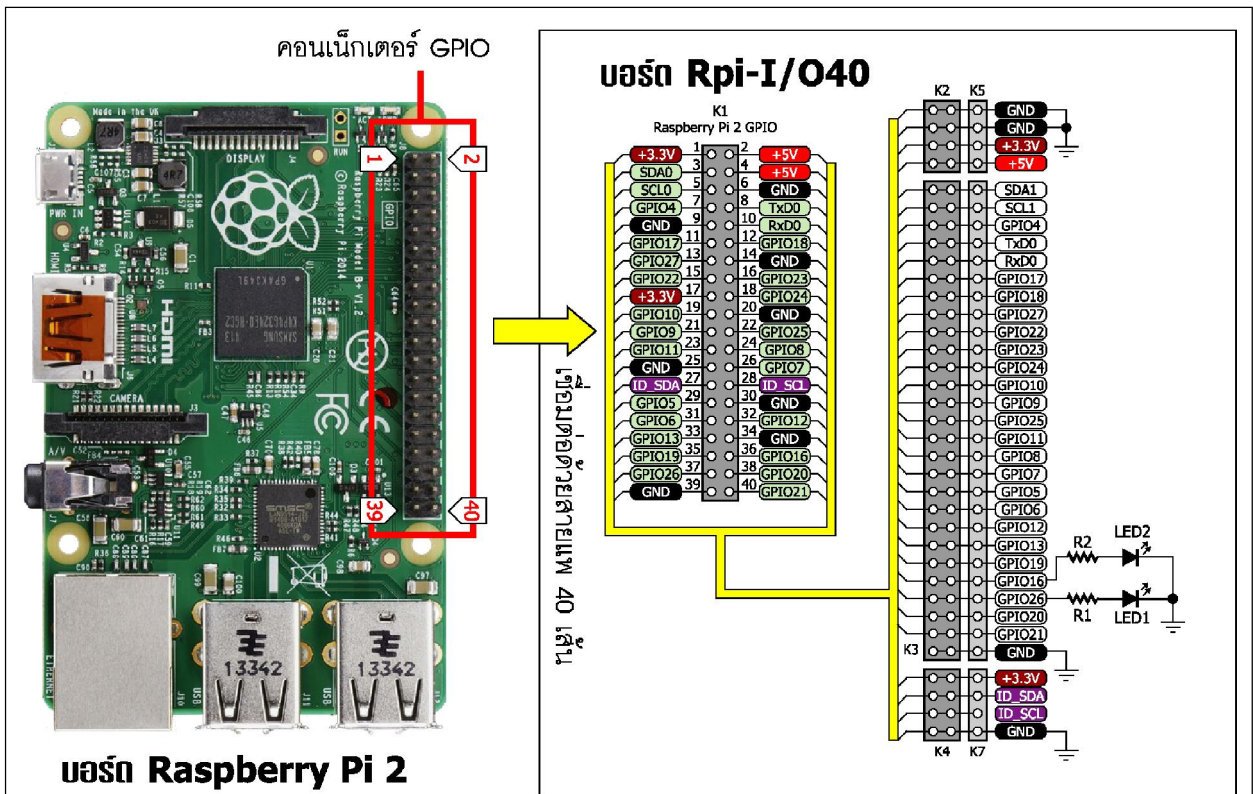
### 12.12.1 ภาพรวมและขั้นตอนการทำงาน

แสดงด้วยไดอะแกรมในรูปที่ 12-24 สรุปเป็นขั้นตอนได้ดังนี้

- (1) เว็บเบราว์เซอร์ (คอมพิวเตอร์) ร้องขอเว็บเพจที่ชื่อว่า **Control.php** ผ่านทางเว็บเบราว์เซอร์ (Internet Explorer หรือ Google Chrome หรือ Mozilla Firefox) มายังเว็บเซิร์ฟเวอร์ (บอร์ด Raspberry Pi 2)
- (2) ไฟล์สคริปต์ **Control.php** จะทำงาน เพื่อสั่งให้พอร์ต GPIO ทำงานตามที่กำหนด
- (3) LED ที่ต่อไว้กับขาพอร์ต GPIO ของ Raspberry Pi 2 ทำงานตามข้อมูลที่ส่งมา
- (4) เมื่อไฟล์สคริปต์ **Control.php** ทำงานเสร็จแล้ว จะส่งเฉพาะเว็บเพจ HTML กลับไปยังเว็บเบราว์เซอร์



รูปที่ 12-24 ไดอะแกรมการทำงานของระบบควบคุมอุปกรณ์เอาต์พุตอย่างง่ายผ่านเว็บเบราว์เซอร์โดยใช้ PHP และ HTML



รูปที่ 12-25 วงจรทดลองที่ต่อเข้ากับพอร์ต GPIO ของบอร์ด Raspberry Pi 2 เพื่อแสดงผลการทำงานของระบบควบคุมอุปกรณ์เอาต์พุตอย่างง่ายผ่านเว็บเบราว์เซอร์โดยใช้ PHP และ HTML

## 12.12.2 ขั้นตอนการทดลอง

(1) ต่อดวงจรมารูปที่ 12-25 ให้กับพอร์ต GPIO ของ บอร์ด Raspberry Pi 2

(2) สร้างไฟล์สคริปต์ด้วยภาษา PHP โดยเปิดโปรแกรม Geany เลือกไปที่ **File > New-(with\_Template) > file.php** พิมพ์คำสั่ง PHP ดังโปรแกรมที่ 12-4 สคริปต์นี้เป็นการรับส่งข้อมูลผ่านเว็บเพจ โดยใช้เมธอด Post ในการรับส่งข้อมูลระหว่างเว็บเพจ สังเกตได้จากคำสั่ง `<form method="post">` ซึ่งเป็นตัวกำหนดว่า form นั้นใช้เมธอดใดในการรับส่งข้อมูล

```
<?php
system("gpio -g mode 16 out");
system("gpio -g mode 26 out");
exec ("gpio -g read 16", $st_LED1);
exec ("gpio -g read 26", $st_LED2);
$st_post=0;
if (isset($_POST['LED1']))
{
    $st_post=1;
    if ($st_LED1[0]==0){system("gpio -g write 16 1");}
    else{system("gpio -g write 16 0");}
}
if (isset($_POST['LED2']))
{
    $st_post=1;
    if ($st_LED2[0]==0){system("gpio -g write 26 1");}
    else{system("gpio -g write 26 0");}
}
exec ("gpio -g read 16", $st_LED1);
exec ("gpio -g read 26", $st_LED2);
?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
<head>
<title>Control</title>
<meta http-equiv="content-type" content="text/html; charset=utf-8" />
<meta name="generator" content="Geany 1.22" />
</head>
<body>
LED1=GPIO16<br>
LED2=GPIO26<br>
<form method="post">
```

โปรแกรมที่ 12-4 ไฟล์ Control.php สคริปต์ภาษา PHP สำหรับติดตั้งลงในเว็บเซิร์ฟเวอร์ (บอร์ด Raspberry Pi 2) เพื่อรับข้อมูลสำหรับควบคุมอุปกรณ์เอาต์พุตอย่างง่ายผ่านเว็บเบราว์เซอร์โดยใช้ PHP และ HTML (มีต่อ)



```

<table style="width:50%;">
  <tbody>
    <tr>
      <td>Click LED </td>
      <td>LED Status </td>
    </tr>
    <tr>
      <td>
        <button name="LED1">
          <?php
            if ($st_LED1[0]==0){echo "LED1 On";}
            else{echo "LED1 Off";}
          ?>
        </button>
      </td>
      <td>
        <?php
          if ($st_LED1[0]==0){echo "LED1 OFF";}
          else{echo "LED1 ON";}
        ?>
      </td>
    </tr>
    <tr>
      <td>
        <button name="LED2">
          <?php
            if ($st_LED2[0]==0){echo "LED2 On";}
            else{echo "LED2 Off";}
          ?>
        </button>
      </td>
      <td>
        <?php
          if ($st_LED2[0]==0){echo "LED2 OFF";}
          else{echo "LED2 ON";}
        ?>
      </td>
    </tr>
  </tbody>
</table>
</form>
</body>
<?php
  if ($st_post==1){echo"<script>>window.location='Control.php'</script>";}
?>
</html>

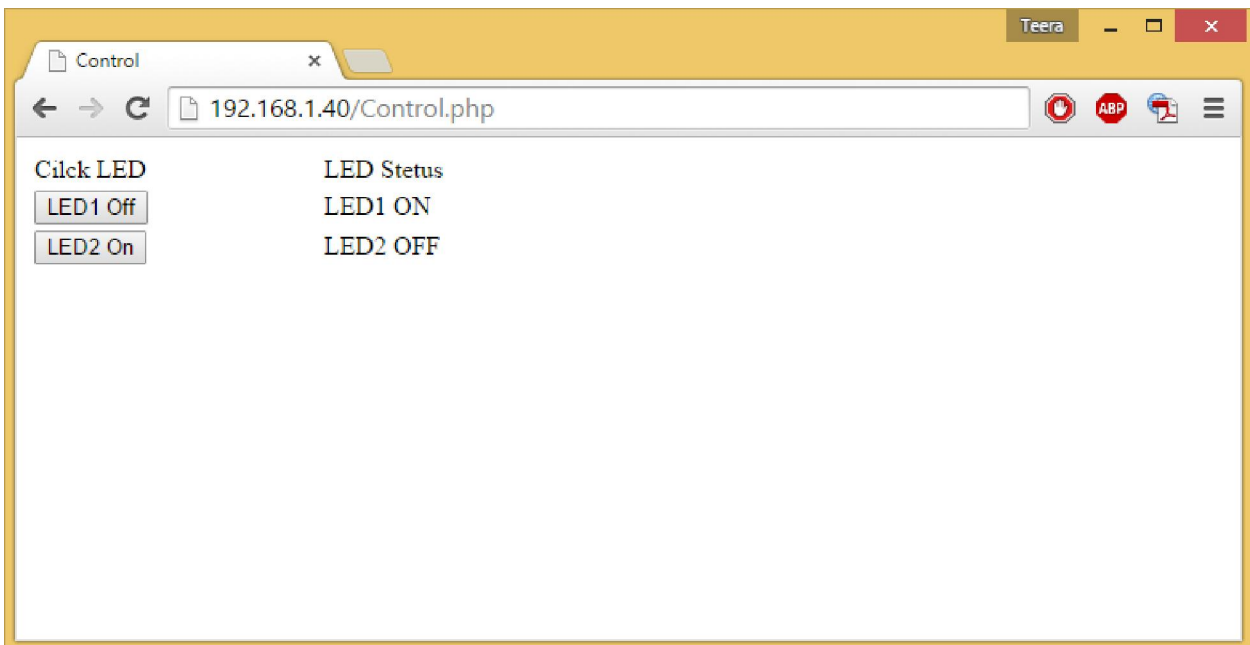
```

โปรแกรมที่ 12-4 ไฟล์ Control.php สคริปต์ภาษา PHP สำหรับติดตั้งลงในเว็บเซิร์ฟเวอร์ (บอร์ด Raspberry Pi 2) เพื่อรับข้อมูลสำหรับควบคุมอุปกรณ์เอาต์พุตอย่างง่ายผ่านเว็บเบราว์เซอร์โดยใช้ PHP และ HTML (จบ)

(3) บันทึกไฟล์ชื่อ **Control.php** เก็บไว้ที่ `/var/www/`

(4) จากนั้นทำการทดสอบการทำงานของสคริปต์ โดยไปที่คอมพิวเตอร์ที่ทำหน้าที่เป็นเว็บไคลเอ็นต์ เปิดเว็บเบราว์เซอร์ให้พิมพ์ IP แอดเดรสของ Raspberry Pi 2 (ได้มาจากหัวข้อก่อนหน้านี้) ตามด้วยชื่อไฟล์ที่สร้างและค่าต่างๆ เช่น `192.168.137.243/Control.php` จะได้ผลการทำงานเป็นหน้าเว็บ (เว็บเพจ) ดังรูปที่ 12-26

(5) คลิกปุ่ม **LED1 on** สถานะ **LED1** จาก **OFF** จะเปลี่ยนเป็น **ON** ทำให้ LED ที่ต่อกับขาพอร์ต GPIO16 ติดสว่าง และปุ่ม **LED1** จะเปลี่ยนเป็น **LED1 off** หากคลิกปุ่มเดิมอีกครั้ง LED ที่ติดอยู่จะดับลง



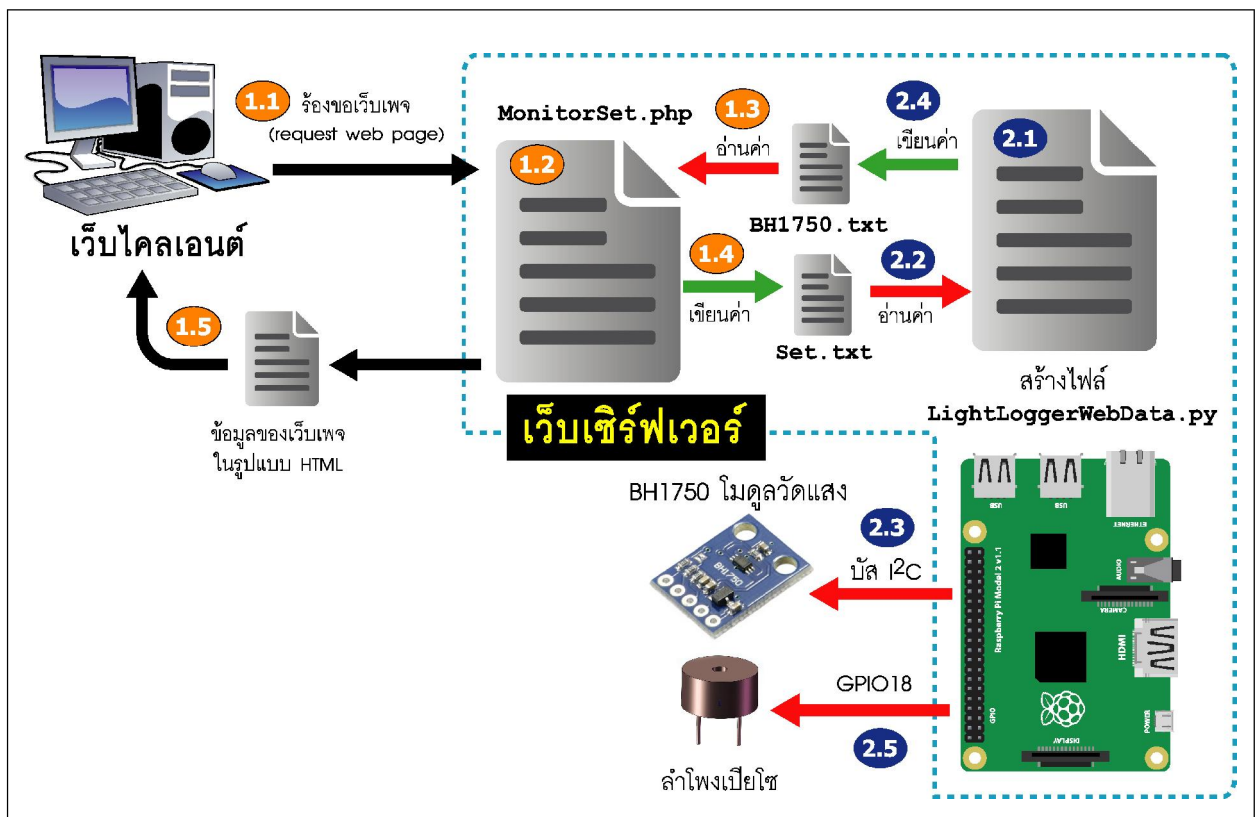
รูปที่ 12-26 เว็บเบราว์เซอร์แสดงเว็บเพจสำหรับควบคุมการทำงานของอุปกรณ์เอาต์พุตผ่านโดยใช้ PHP และ HTML

## 12.13 การทดลองติดต่ออุปกรณ์อินพุตเอาต์พุตผ่านเว็บเบราว์เซอร์ ทำงานร่วมกับไฟล์สคริปต์ Python อย่างง่าย

การติดต่อเพื่อควบคุมอุปกรณ์ผ่านเครือข่ายของบอร์ด Raspberry Pi 2 นอกจากการใช้ไฟล์สคริปต์ที่พัฒนาขึ้นด้วยภาษา PHP ทำงานร่วมกับ HTML แล้ว ยังมีอีกแนวทางหนึ่งที่ขอนำเสนอในหัวข้อนี้ นั่นคือ การใช้ไฟล์สคริปต์ที่พัฒนาด้วยภาษา Python ทำงานร่วมกับเว็บเบราว์เซอร์และไฟล์สคริปต์ที่พัฒนาด้วยภาษา PHP เพื่อติดต่อกับอุปกรณ์อินพุตเอาต์พุตที่ต่อกับพอร์ต GPIO ของบอร์ด Raspberry Pi 2 ซึ่งได้รับการกำหนดให้ทำงานเป็นเว็บเซิร์ฟเวอร์

### 12.13.1 ภาพรวมและขั้นตอนการทำงาน

ในรูปที่ 12-27 แสดงไดอะแกรมการทำงานในภาพรวมของการติดต่ออุปกรณ์อินพุตเอาต์พุตผ่านเว็บเบราว์เซอร์ซึ่งทำงานร่วมกับไฟล์สคริปต์ Python สรุปขั้นตอนการทำงานได้ดังนี้



รูปที่ 12-27 ไดอะแกรมการทำงานของระบบควบคุมผ่านเครือข่ายที่ใช้บอร์ด Raspberry Pi 2 เป็นเว็บเซิร์ฟเวอร์ โดยมีการเชื่อมต่อกับอุปกรณ์อินพุตเอาต์พุตที่ต้องการอ่านค่าและสั่งการผ่านพอร์ต GPIO

### (1) การทำงานของไฟล์สคริปต์ PHP

- (1.1) เว็บไคลเอนต์ (คอมพิวเตอร์) ร้องขอเว็บเพจที่ชื่อว่า **MonitorSet.php** ผ่านทางเว็บเบราว์เซอร์ไปยังเว็บเซิร์ฟเวอร์ (บอร์ด Raspberry Pi 2)
- (1.2) ไฟล์สคริปต์ **MonitorSet.php** ทำงาน
- (1.3) อ่านไฟล์ **BH1750.txt** เพื่อนำค่ามาแสดงผล
- (1.4) บันทึกค่าลงบนไฟล์ **set.txt** ในกรณีที่ต้องการตั้งค่าใหม่
- (1.5) เมื่อไฟล์สคริปต์ **MonitorSet.php** ทำงานเสร็จแล้ว จะส่งเฉพาะเว็บเพจ HTML กลับไปยังเว็บไคลเอนต์

### (2) การทำงานของไฟล์สคริปต์ Python

- (2.1) ทำการเอ็ชคิวต์หรือสั่งให้ไฟล์สคริปต์ที่ชื่อว่า **LightLoggerWebData.py** ทำงาน
- (2.2) อ่านไฟล์ **set.txt** เพื่อกำหนดการแจ้งเตือน
- (2.3) อ่านค่าจากตัวตรวจจับและวัดค่าความเข้มของแสงในหน่วยลักซ์ (lux)
- (2.4) บันทึกค่าที่ได้จากตัวตรวจจับแสงลงในไฟล์ **BH1750.txt**
- (2.5) ส่งสัญญาณออกขา GPIO18 เพื่อแจ้งเตือน

ในตัวอย่างการทดลองที่จะได้เรียนรู้ไปพร้อมกันนี้ ใช้ไฟล์ **set.txt** เป็นตัวกลางในการติดต่อสื่อสารระหว่าง สคริปต์ที่ใช้แสดงเป็นหน้าเว็บ กับสคริปต์ Python ที่ควบคุมอุปกรณ์

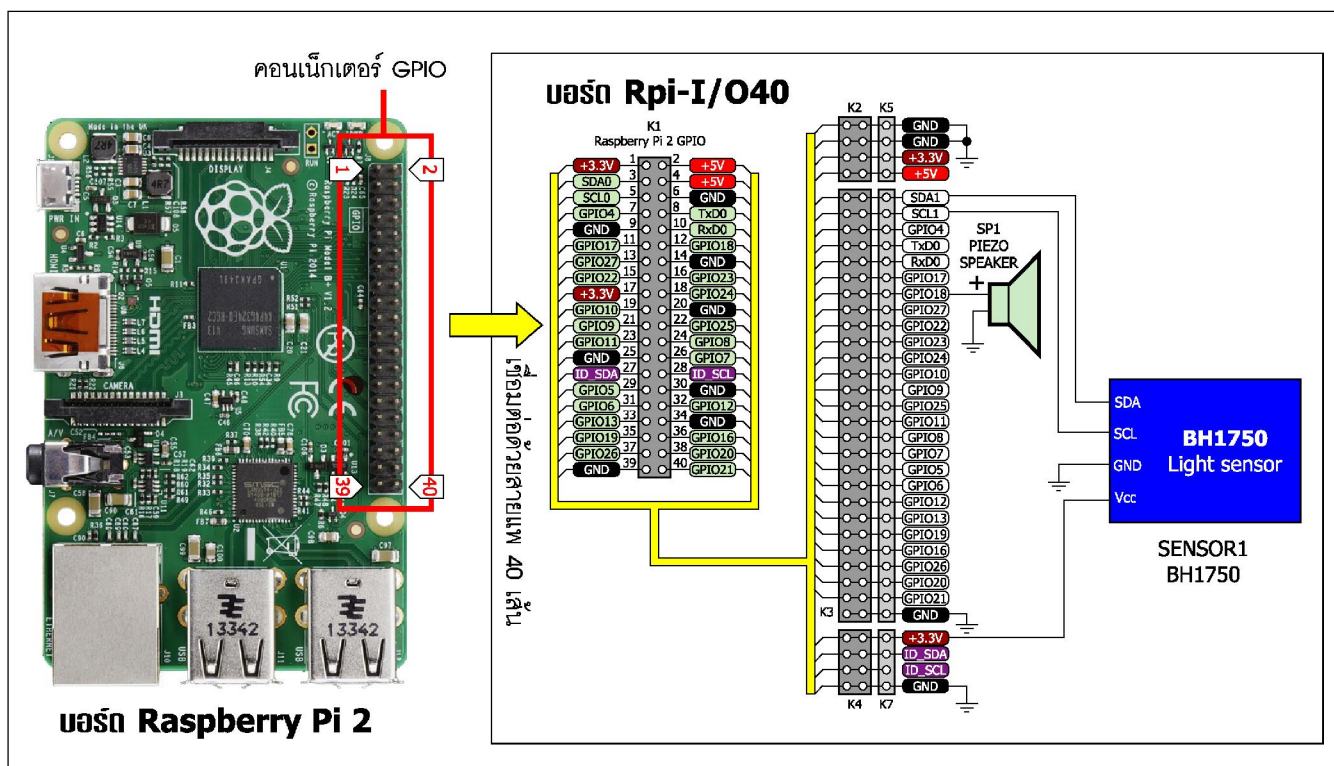
## 12.13.2 ขั้นตอนการทดลอง

(1) ต่อวงจรตามรูปที่ 12-28 โดยต่อกับบอร์ด Raspberry Pi 2 ผ่านทางพอร์ต GPIO วงจรทดลองที่ต่อเข้ากับพอร์ต GPIO ของบอร์ด Raspberry Pi 2 เพื่อแสดงผลการทำงานของระบบควบคุมอุปกรณ์ เอาต์พุตอย่างง่ายผ่านเว็บเบราว์เซอร์โดยใช้ PHP และ HTML

(2) ที่บอร์ด Raspberry Pi 2 ทำการเปิดโปรแกรม Geany เพื่อนำไปสร้างสคริปต์ไฟล์ Python แล้วตั้งชื่อว่า **LightLoggerWebData.py**

(3) ทำการบันทึกไฟล์

(4) ลำดับต่อไปสร้างเท็กซ์ไฟล์ (ไฟล์ .txt) กำหนดให้มีชื่อว่า **set.txt** สำหรับรับข้อมูลเฉพาะค่าตัวเลขเข้ามา บันทึกไว้ในไดเรกทอรี **/var/www/**



รูปที่ 12-28 จอทรทดลองที่ต่อเข้ากับพอร์ต GPIO ของบอร์ด Raspberry Pi 2 เพื่อติดต่อกับอุปกรณ์อินพุตเอาต์พุตอย่างง่ายผ่านเว็บเบราว์เซอร์โดยใช้ PHP และ Python

(6) สร้างไฟล์สคริปต์ PHP เพื่อแสดงหน้าเว็บเพจ

(6.1) เปิดโปรแกรม Geany สร้างไฟล์สคริปต์ PHP โดยไปที่ **File>>New(with\_Template)->>file.php** แล้วตั้งชื่อว่า **MonitorSet.php**

(6.2) เพิ่มสคริปต์ PHP และ javascript ดังโปรแกรมที่ 12-6

```

import RPi.GPIO as GPIO
import time
import smbus
import datetime
GPIO.setwarnings(False)
GPIO.setmode(GPIO.BCM)
GPIO.setup(18, GPIO.OUT)
blink = GPIO.PWM(18,500)
st=0
blink.start(0)
bus = smbus.SMBus(1) #(512MB)
addr = 0x23 # i2c adress
dataset=50
while True:
    try:
        f=open('/var/www/set.txt','r')
        dataset=float(f.read())
    except:
        pass
    date=datetime.datetime.now()
    microsec=date.microsecond
    if microsec > 700000:
        blink.ChangeDutyCycle(st)
    else:
        blink.ChangeDutyCycle(0)
    data = bus.read_i2c_block_data(addr,0x11)
    lum=(data[1] + (data[0]<<8) / 1.2)
    with open("/var/www/BH1750.txt", "w") as text_file:
        text_file.write("DateTime: %s<br>Luminosity: %.2f lx<br>" % (date,lum,))
    if lum<dataset:
        st=50
    else:
        st=0
    time.sleep(0.2)

```

โปรแกรมที่ 12-5 ไฟล์ `LightLoggerWebData.py` เป็นโปรแกรมภาษา Python สำหรับติดต่อกับบอร์ด Raspberry Pi 2 ผ่านทางพอร์ต GPIO เพื่ออ่านค่าแสงกลับไปแสดงผลและรับข้อมูลเพื่อนำไปขับเสียงออกลำโพงเปียโซ

```

<?php
  if ((isset($_POST['textbox'])) & ($_POST['textbox'] != ""))
  {
    $fp=fopen('set.txt','w');
    fwrite($fp,$_POST['textbox']);
    fclose($fp);
  }

  $frp=fopen('set.txt','r');
  $rdata=fgets($frp);
  fclose($frp);
?>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
  <head>
    <title>MonitorSet</title>
    <meta http-equiv="content-type" content="text/html; charset=utf-8" />
    <meta name="generator" content="Geany 1.22" />
    <script language="JavaScript" type="text/javascript">
      var rFile;
      function Call()
      {
        if(rFile.readyState == 4) // ตรวจสอบว่าจบการทำงานหรือไม่
        {
          if(rFile.status==200) // ตรวจสอบว่าเชื่อมต่อกับเซิร์ฟเวอร์ได้หรือไม่
          {
            var allText = rFile.responseText; // นำค่าที่ส่งกลับมาเก็บไว้ในตัวแปร
            if(allText!="") // ตรวจสอบว่ามีข้อความหรือไม่
            {
              document.getElementById("myDiv").innerHTML = allText;
              // แสดงข้อความที่ได้
            }
            setTimeout("readTextFile()",500); // รอได้รีเฟรชทุกๆ 0.5 วินาที
          }
        }
      }
    }
  }
}

```

โปรแกรมที่ 12-6 ไฟล์ MonitorSet.php เป็นไฟล์สคริปต์ PHP สำหรับสร้างเว็บเพจของเว็บเซิร์ฟเวอร์ (บอร์ด Raspberry Pi 2) เพื่อแสดงค่าและสั่งการอุปกรณ์อินพุตเอาต์พุตผ่านเว็บเบราว์เซอร์ (มีต่อ)

```

function readTextFile()
{
    rFile = new XMLHttpRequest(); // สร้าง object XMLHttpRequest
    rFile.open("GET","BH1750.txt",true);
    // เปิดการติดต่อไฟล์ BH1750.txt กับเว็บเซิร์ฟเวอร์
    rFile.onreadystatechange = Call;
    // เรียกใช้ฟังก์ชัน call เมื่อมีการเปลี่ยนแปลงสถานะการทำงาน
    rFile.send(null); // ทำการส่งข้อมูล
};

</script>
</head>

<body onload="readTextFile()" >
    <form method="post">
        <div id="myDiv"></div>
        <? echo "Set Alarm = ".$rdata." lx <br>"?>
        <tr>
            Set Alarm:
            <td><input type="text" name="textbox" ></td>
            <td><button name="OK">OK</button></td>
        </tr>
    </form>
</body>
</html>

```

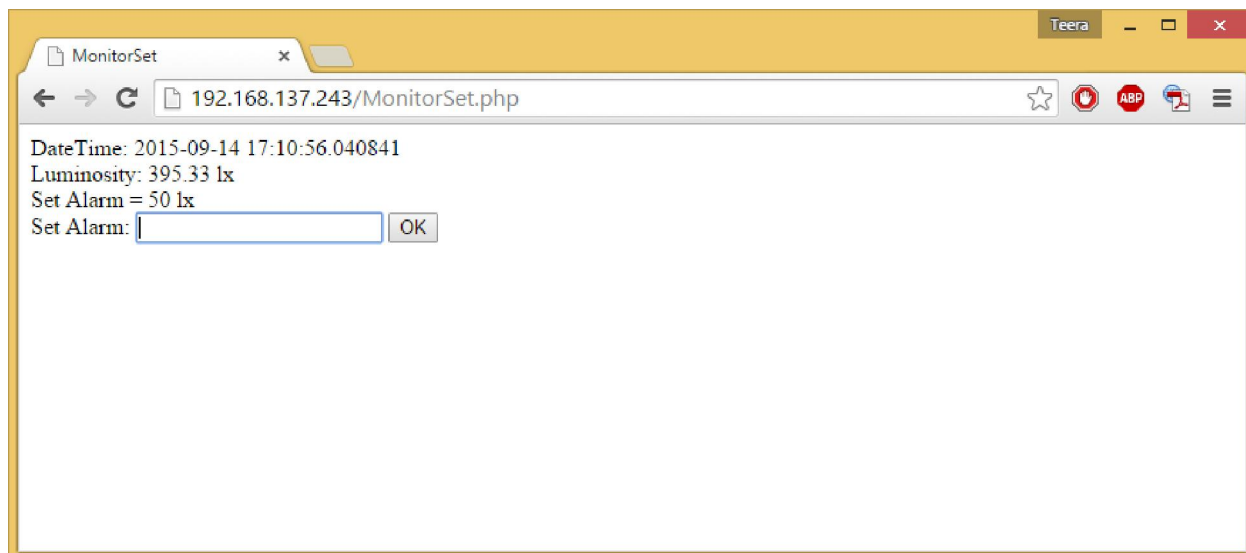
### การทำงานของโปรแกรม

ในกรณีเมื่อกดปุ่ม OK รับค่าจาก Textbox ด้วยตัวแปร `$_POST['textbox']` จะทำการตรวจสอบว่ามีค่าหรือไม่ ถ้ามีจะบันทึกลงในไฟล์ `set.txt` จากนั้นอ่านค่าจากไฟล์ `set.txt` เก็บไว้ในตัวแปร `$rdata` เพื่อนำไปแสดงผล ส่วนสคริปต์ javascript ทำหน้าที่อ่านข้อมูลทุกๆ 0.5 วินาทีจากไฟล์ `HB1750.txt` มาแสดงผลอย่างต่อเนื่อง โดยไม่ต้องรีเฟรชหน้าเว็บเอง หรือเรียกอีกอย่างคือออโต้รีเฟรช (auto refresh) และยังสามารถทำงานอื่นได้โดยไม่ต้องรอ เรียกกระบวนการทำงานแบบนี้ว่า *AJAX (Asynchronous JavaScript and XML)*

สำหรับไฟล์ `HB1750.txt` จะถูกสร้างขึ้นเมื่อสั่งให้สคริปต์ `LightLoggerWebData.py` ทำงาน

**โปรแกรมที่ 12-6 ไฟล์ `MonitorSet.php` เป็นไฟล์สคริปต์ PHP สำหรับสร้างเว็บเพจของเว็บเซิร์ฟเวอร์ (บอร์ด Raspberry Pi 2) เพื่อแสดงค่าและสั่งการอุปกรณ์อินพุตเอาต์พุตผ่านเว็บเบราว์เซอร์ (จบ)**





รูปที่ 12-29 เว็บเบราว์เซอร์แสดงเว็บเพจสำหรับอ่านค่าและกำหนดการทำงานของอุปกรณ์อินพุตเอาต์พุตผ่านพอร์ต GPIO ของบอร์ด Raspberry Pi 2 ซึ่งทำหน้าที่เป็นเว็บเซิร์ฟเวอร์โดยใช้ PHP และไฟล์สคริปต์ที่พัฒนาจากโปรแกรมภาษา Python

### 12.13.33 การทดสอบ

- (1) เริ่มต้นด้วยการเอ็กซ์คิวิตไฟล์สคริปต์ที่ชื่อ **LightLoggerWebData.py** ก่อน
- (2) ไปที่คอมพิวเตอร์ที่ทำหน้าที่เป็นเว็บไคลเอ็นต์ เปิดเว็บเบราว์เซอร์ให้พิมพ์ IP แอดเดรสของ Raspberry Pi 2 ตามด้วยชื่อไฟล์ที่สร้างและค่าต่างๆ เช่น **192.168.137.243/MonitorSet.php** จะแสดงเว็บเพจดังรูปที่ 12-29 จากการทดสอบพบว่า ใช้ *Google Chrome* จะทำงานได้สมบูรณ์แบบที่สุด
- (3) ใช้มือบังแสงที่ตกกระทบโมดูลตรวจจับแสง BH1750 เพื่อทำให้วัดค่าความเข้มแสงได้น้อยกว่า 50 จะทำให้บอร์ด Raspberry Pi 2 ขับสัญญาณเสียงเตือนให้ดังขึ้นผ่านลำโพงเปียโซ
- (4) ผู้ทดลองสามารถตั้งค่าแจ้งเตือนใหม่ได้ โดยกรอกค่าลงในช่อง **SetAlarm** แล้วกดปุ่ม **OK** ค่าที่กำหนดใหม่จะถูกเก็บไว้ในไฟล์ **sex.txt** ในไดเรกทอรี **/var/www**

ทั้งหมดที่นำเสนอในบทนี้เป็นตัวอย่างการพัฒนาระบบควบคุมผ่านเครือข่ายอย่างง่ายกับบอร์ด Raspberry Pi 2 ซึ่งกำหนดให้ทำหน้าที่เป็นเว็บเซิร์ฟเวอร์มีการติดต่อกับอุปกรณ์อินพุตเอาต์พุตผ่านพอร์ต GPIO หากการควบคุมอุปกรณ์ไม่ซับซ้อนมากนัก การใช้ไฟล์สคริปต์ PHP ทำงานร่วมกับ HTML จะเป็นทางเลือกที่ใช้ได้เมื่อมีความต้องการติดต่อกับอุปกรณ์อินพุตเอาต์พุตที่ซับซ้อนขึ้น การพัฒนาโปรแกรมด้วย Python จะตอบสนองได้ดีกว่าเมื่อผู้พัฒนา มีความเข้าใจในการทำงานโดยรวมของระบบแล้ว ก็สามารถต่อยอดเพื่อพัฒนาเว็บเพจของเว็บเซิร์ฟเวอร์ให้มีหน้าตาที่สวยงามขึ้นได้ หากแต่ต้องมีความเข้าใจในข้อจำกัดด้านทรัพยากรของบอร์ด Raspberry Pi 2 ด้วย เพื่อป้องกันไม่ให้งานโดยรวมช้าเกินไป

