

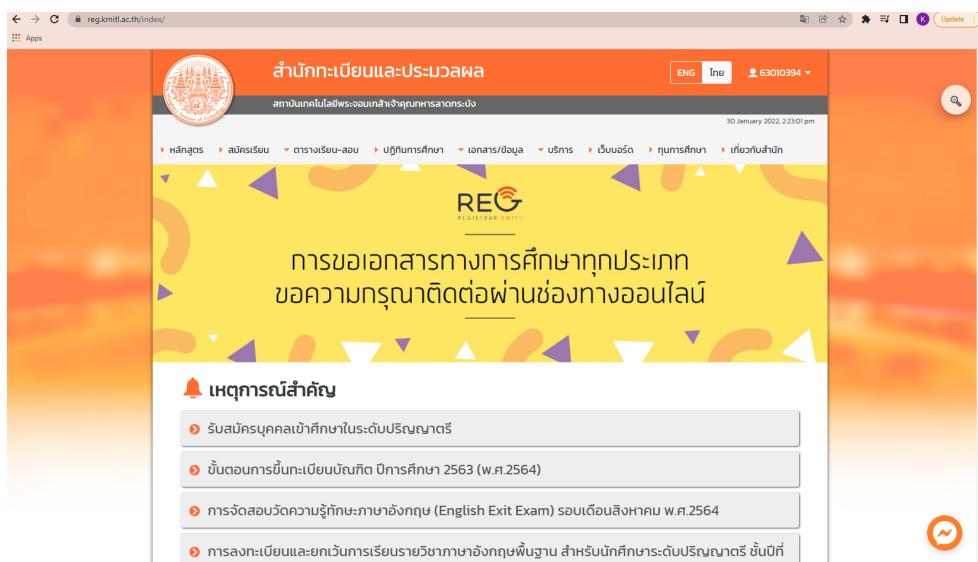
# Table of Contents

<b>Table of Contents</b>	<b>1</b>
<b>Website</b>	<b>3</b>
ประวัติของ Website	4
<b>Website working</b>	<b>5</b>
โครงสร้าง HTML	7
HTTP	11
HTTP request	12
ส่วนประกอบของ Request Message	13
Request-Line	13
Request Method	13
Request Header Fields	14
Message-body	14
HTTP response	14
ส่วนประกอบของ Response Message	15
Status-Code และ Reason-Phrase	16
Response Header Fields	16
Message-body	16
Catch/Cookies	17
HTTP With SSL	17
<b>Inspector 101</b>	<b>22</b>
Overview	22
Toolbar	28
Element	32
Filter	34
Console	36
Sources	40
Network	41
Application	44
Security	45
<b>Activity CTF</b>	<b>47</b>
PicoCTF register	47
<b>ภาคผนวก</b>	<b>49</b>

ໄວເປ່ອຮລິງກໍ	49
ສຕາບນ້າເຊີຣນ (CERN)	49
ໄຄລເອັນຕົ - ເຊື້ອົມເວຼອຮ (Client - Server)	50
just-in-time compilation	50
Rules of Request Methods	51
Request for Comments (RFC)	52
WebDAV	53
XML (Extensible Markup Language)	53
Nginx	54
Status Code ແລະ Reason Phrase ເພີ່ມເຕີມ	54

## Website

ເວັບໄຊຕ່າງໆ ນມາຍຄົງໜ້າເວັບເພຈລາຍໜ້າ ຜົ່ງເຊື່ອມໂຍງກັນຜ່ານກາງ **ໄຣເປົກລົງກ** ສ່ວນໃໝ່ຈະດຳກຳຂຶ້ນເພື່ອນໍາເສນວ່າຂໍ້ມູນຜ່ານຄວມພິວເຕອນ ໂດຍຄູກຈັດເກີບໄວ້ໃນເວີලຕົວໄວ້ເວັບໄຊຕ່າງໆ ແກ້ໄຂໃຫ້ເກີບໄວ້ທີ່ຂໍ້ອ້າລັກຈະເຮັຍກ່າວ່າ ໂຮມເພຈ ເວັບໄຊຕ່າງໆໄດ້ກ່າວ່າໄປຈະໃຊ້ບໍລິການເຕືອນໄຫວ້າຜູ້ໃຊ້ພົມ ແຕ່ໃນຂໍ້ມູນເດີຍວັກນບາງເວັບໄຊຕ່າງໆຈໍາເປັນຕ້ອງມີການສ່ວນມືກະສານີກແລະເສີຍຄ່າບໍລິການເພື່ອທີ່ຈະດູຂໍ້ມູນໃນເວັບໄຊຕ່າງໆເລຳລ່ານັ້ນ ຍກຕັວອ່າງເຈັ່ງເຊັ່ນຂໍ້ມູນກາງວິຊາການ ຂໍ້ມູນຕາລາດໍາລັກກຮຽນ ຂອບຂໍ້ມູນລ້ົ່ວຕ່າງໆ ຖ້າຜູ້ກຳເວັບໄຊຕ່າງໆມີລາກຫາຍະຮະດັບ ຕັ້ງແຕ່ສ້າງເວັບໄຊຕ່າງໆສ່ວນຕົ້ນ ຈົນຄົງຮະດັບເວັບໄຊຕ່າງໆສໍາໜັບຮູຖາກີຈ ຂອງອົງຄ່າກົດຕ່າງໆ ຈົນການເຮັດວຽກໃຫ້ຕ່າງໆໄດ້ກ່າວ່າໄປນິຍມເຮັດວຽກ ຜ່ານໜອົວໃຫ້ຕ່າງໆໃນລັກຈະນະຂອງເວັບເປົກລົງ



ຮູບກີ່ 1 ຕົວອ່າງເຈັ່ງເວັບໄຊຕ່າງໆ

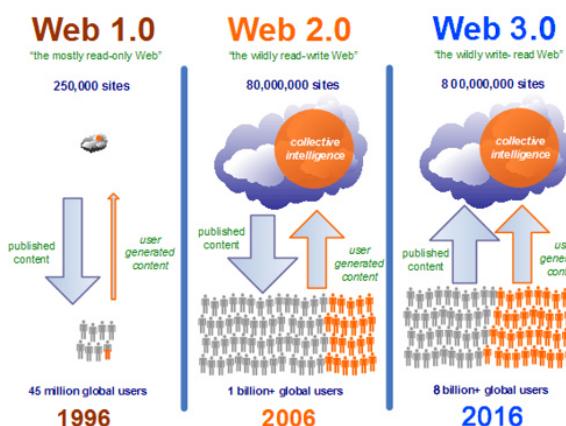
## ประวัติของ Website

ในปี พ.ศ.2533 นักวิทยาศาสตร์จากห้องทดลองของ **สถาบันเชร์น (CERN)** คือ ทิม เบอร์นเนอร์ส-ลี (Tim Berners-Lee) ได้สร้างระบบการสื่อสารข้อมูลผ่านเครือข่ายคอมพิวเตอร์ในรูปแบบใหม่ ที่เรียกว่า ไฮเพอร์เทกซ์ (Hypertext) ซึ่งผลก็ได้ทำให้มีการสร้าง协议โพรโท콜แบบ HTTP (Hypertext Transfer Protocol) ขึ้น เพื่อใช้ในการส่งสารสนเทศต่าง ๆ โดยจะถูกจัดอยู่ในรูปแบบใหม่ที่เรียกว่า HTML (HyperText Markup Language) ซึ่งการสื่อสารและการสืบค้นสารสนเทศในรูปแบบใหม่นี้ทำให้มีความสามารถติดต่อสื่อสารกันได้อย่างรวดเร็วในทุกรูปแบบ ไม่ว่าจะเป็นข้อความ ภาพ และเสียงซึ่งด้วยตัวของเว็บไซต์นั้นก็ได้พัฒนามาอย่างต่อเนื่องโดยจะแบ่งได้เป็น 3 ยุคใหญ่ ๆ ดังนี้

ยุคที่ 1 Static Web (Web 1.0) เป็นการเขียนเว็บโดยใช้ภาษา HTML ล้วน ๆ อย่างมากที่มีการใช้สคริปต์ฝั่งไคลเอนต์ เช่น JavaScript VBScript หรือ Java Applet ส่วนใหญ่จะนิยมใช้ในหน้าแรกเรียนและใช้สร้างโฮมเพจส่วนตัว บทความทางวิชาการ เป็นต้น โดยที่ส่วนใหญ่แล้วจะเป็นเว็บไซต์ที่สามารถอ่านได้อย่างเดียว

ยุคที่ 2 Dynamic Web (Web 2.0) ได้รับการพัฒนามาจากยุคที่ 1 มีการใช้สคริปต์ทางฝั่งเซิร์ฟเวอร์ มาช่วยในการเพิ่มความสามารถของ HTML ในการติดต่อกันระหว่างเซิร์ฟเวอร์ อย่างฐานข้อมูล หรืออาศัยความสามารถในการประมวลผลของเว็บเซิร์ฟเวอร์เพื่อทำงานบางอย่าง เช่น Search Webboard Guestbook Chat room Webmail โดยที่เทคโนโลยีในการพัฒนาเว็บในช่วงแรกคือ CGI ภาษาที่ใช้เขียนสคริปต์นี้ เช่น C Perl ต่อมาได้มีการพัฒนาเทคโนโลยีที่มีการทำงานลักษณะแบบตัวต่อตัว เช่น ASP PHP JSP เป็นต้น

ยุคที่ 3 Web services (Web 3.0) เป็นรูปแบบบริการยุคใหม่ในวงการเว็บ ตัวอย่าง Web Service ที่เห็นเด่นชัด เช่น Microsoft Passport ที่ใช้บริการตรวจสอบความเป็นตัวตนจริง ผ่านเว็บภาษาที่ใช้เป็นตัวกลางในการพัฒนาเว็บเชอร์ฟิล์ดคือ XML นั่นเอง IBM ได้นิยามความหมายอย่างเป็นทางการของ Web Service ว่า เว็บเซอร์ฟิล์ด คือ Web Application ยุคใหม่ที่ประกอบด้วยส่วนย่อย ๆ มีความสมบูรณ์ในตัวเอง สามารถติดต่อกัน ค้นหา เริ่มทำงานได้ผ่านเว็บ Web Service สามารถทำงานได้ตั้งแต่งานง่าย ๆ เช่น ดึงข้อมูล จนถึงกระบวนการทางธุรกิจที่ซับซ้อน เมื่อ Web Service ตัวใดตัวหนึ่งเริ่มทำงาน Web Service ตัวอื่นก็สามารถรับรู้และเริ่มทำงานได้อีกด้วย



รูปที่ 2 ภาพประกอบวิวัฒนาการของเว็บไซต์

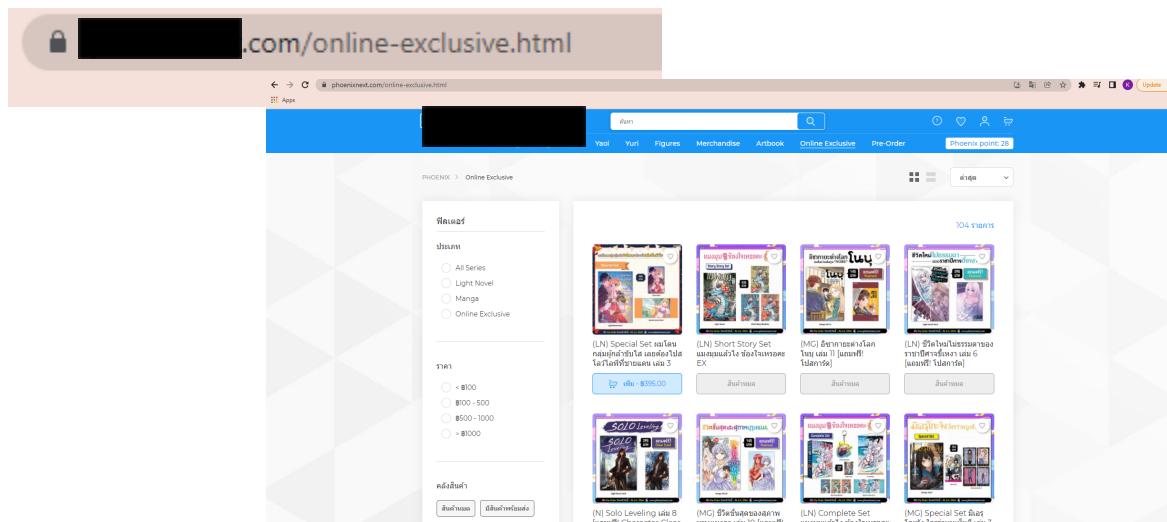
ໂໄຍກີໃນປັຈລຸບນີ້ໄດ້ມີ Web 4.0 ເພີ່ມຂຶ້ນມາດ້ວຍແລ້ວ ໂດຍຮາຍລະເວີຍດີເປັນດັ່ງນີ້

ຢູ່ຄົກທີ 4 A Symbiotic web (Web 4.0) ເປັນເວັບທີ່ກຳຈານແບບ Artificial Intelligence (AI) ກີ່ຈະລາມາມາຍິ່ງຈຶ່ນ ດອມພົວເຕັນສາມາຮັດດີໄດ້ ມີຄວາມຈະລາມາມາຈຶ່ນ ໃນການອ່ານກັ້ງເນື້ອຫາ ຂ້ອຄວາມ ແລະ ຮັບພາບຮອບໃຈໂລ ສາມາຮັດທີ່ຈະຕອບສອນອງຂອງຕັດສິນໃຈໄດ້ວ່າຈະ load ຊ້ອມລວະໄຣມາຈາກທີ່ໃຫນຈິງຈະໃຊ້ປະສົກທີ່ກາພດີທີ່ສຸດມາໃຫ້ຜູ້ໃຊ້ຈານກ່ອນກ່ອນ ແລະນອກຈານນີ້ຢ່າງມີຮູບແບບການນຳມາແສດງທີ່ຮວດເຮົວ ເວັບ 4.0 ຈະກຳໃໝ່ເວັບ ຂອງຊ້ອມລູ່ຕ່າງ ຈາ ສາມາຮັດກຳຈານໄດ້ແກບຈະຖຸກອຸປະກັນໜີ້ອາຈະຈ່ວຍຮະບຸຕັ້ງຕັນທີ່ແກ່ຈົງຂອງຜູ້ໃຊ້ໄດ້ນັ້ນເວັບ

## Website working

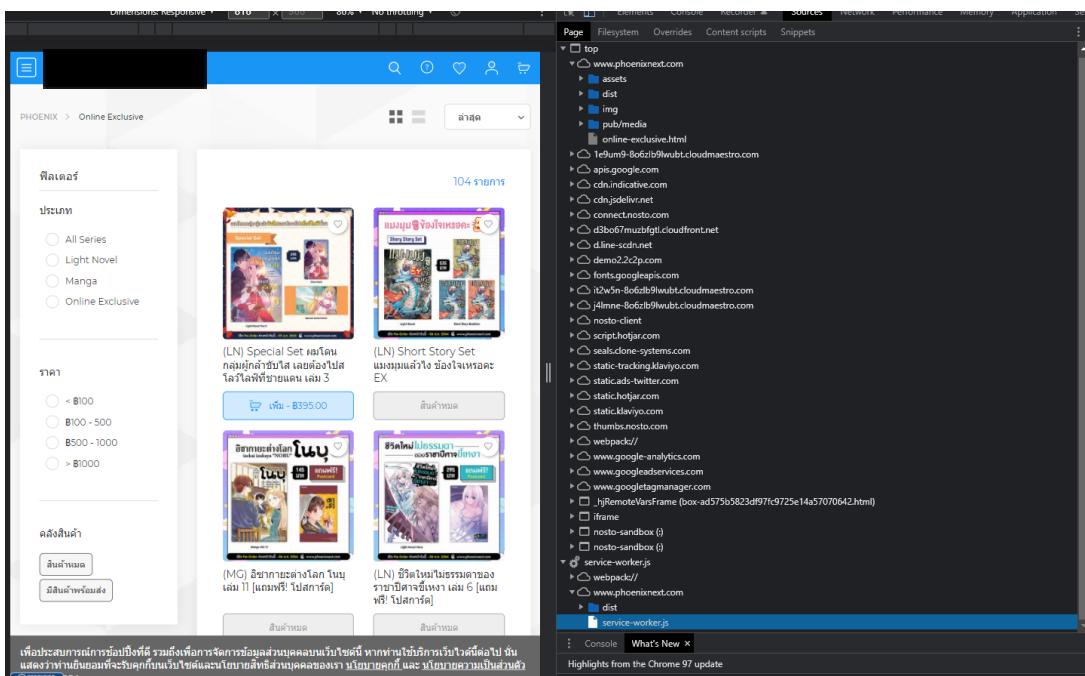
หลักการทำงานของเว็บไซต์จะมีลักษณะ เช่น เดียว กับบริการอื่น ๆ ของอินเทอร์เน็ต คืออยู่ในรูปแบบ **ไคลเอนต์ - เซิร์ฟเวอร์ (Client - Server)** โดยมีโปรแกรมเว็บไคลเอนต์ (Web Client) กำหนดนำที่ เป็นผู้รับรองขอบริการ และมีโปรแกรมเว็บไซต์ (Web Server) กำหนดที่เป็นผู้ให้บริการ โปรแกรมเว็บไคลเอนต์ก็คือโปรแกรมเว็บบราวเซอร์ (Web Browser) นั่นเอง สำหรับโปรแกรมเว็บเซิร์ฟเวอร์นั้นจะถูก ติดตั้งไว้ในเครื่องของผู้ให้บริการเว็บไซต์ การติดต่อระหว่างโปรแกรมเว็บบราวเซอร์กับโปรแกรมเว็บเซิร์ฟเวอร์จะกระทำผ่านโปรโตคอล HTTP (Hypertext Transfer Protocol)

กลไกการทำงานของเว็บเพจสำหรับเว็บเพจหรือรวมๆกับคิมีนามสกุลของไฟล์เป็น HTM หรือ HTML นั้น เมื่อเราใช้เว็บбраузரเปิดดูเว็บเพจได้ เว็บเซิร์ฟเวอร์ก็จะส่งเว็บเพจนั้นกลับยังบราวเซอร์ จากนั้นบราวเซอร์จะแสดงผลไปตามคำสั่งภาษา HTML (Hypertext Markup Language) ที่อยู่ในไฟล์จะเห็นได้ว่าเว็บเพจที่มีลักษณะ Static คือ ผู้ใช้จะพบกับเว็บเพจนานาเดิม ๆ ทุกครั้งจนกว่าผู้ดูแลเว็บ เพจนั้นจะทำการปรับปรุงเว็บเพจนั้น อันนี้คือข้อจำกัดที่มีต้นเหตุมาจากการภาษา HTML สามารถกำหนดให้เว็บเพจมีเนื้อร่างอย่างไรได้ แต่ไม่ช่วยให้เว็บเพจมี “ความฉลาด” ได้



รูปที่ 3 ตัวอย่างหน้าเว็บเพจที่เป็น HTML

## การสร้างเว็บเพจที่มีความคลาดสามารถทำได้หลายวิธีด้วยกัน หนึ่งในนั้นคือการฝังสคริปต์หรือชุดคำสั่งที่ทำงานของฝั่งเซิร์ฟเวอร์ (server – side script) ไว้ในเว็บเพจ



รูปที่ 4 สคริปต์ที่ฝังไว้ในเว็บเพจ

จากที่เราเป็นการทำงานของเว็บเพจที่ฝังสคริปต์ภาษา Javascript ไว้ (ขอเรียกว่าไฟล์ Js) เมื่อเว็บбраузரร้องขอไฟล์ Js ไฟล์ใด เว็บเซิร์ฟเวอร์จะเรียก JavaScript engine ขึ้นมาแปล (Interpret) และประมวลผลคำสั่งที่อยู่ในไฟล์ Js นั้น (แต่ในปัจจุบันเนินจีนอาจจะไม่ได้แปลแต่ประมวลผลเลย [*just-in-time compilation*]) โดยอาจมีการดึงข้อมูลจากฐานข้อมูล หรือเขียนข้อมูลลงไปยังฐานข้อมูล ด้วย หลังจากนั้นผลลัพธ์ในรูปแบบ HTML จะถูกส่งกลับยังบราวเซอร์ บราวเซอร์ก็จะแสดงผลคำสั่ง HTML ที่ได้รับมา ซึ่งจะไม่มีคำสั่ง Js ใด ๆ หลงเหลืออยู่เลยเนื่องจากถูกแปลและประมวลผลโดย JavaScript engine ที่ฝังเซิร์ฟเวอร์ไปหมดแล้ว

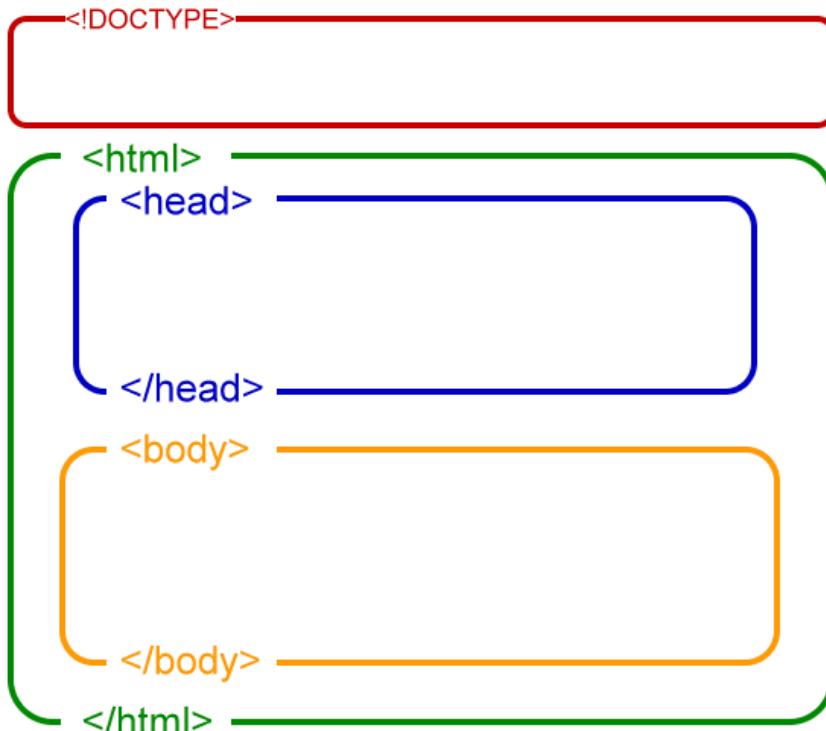
ให้ลองเกตว่าสิ่งที่บราวเซอร์ต้องกระทำการ คือ การร้องขอไฟล์จากเว็บเซิร์ฟเวอร์ จากนั้นก็รอรับผลลัพธ์กลับมาแล้วแสดงผลความแตกต่างจะting ๆ อยู่ที่การทำงานของฝั่งเซิร์ฟเวอร์ ซึ่งกรณีนี้เว็บเพจที่เป็นไฟล์ Js จะผ่านการประมวลผลก่อน แทนที่จะถูกส่งไปยังบราวเซอร์กันที

การฝังสคริปต์ Js ไว้ในเว็บเพจ ช่วยให้เราสร้างเว็บเพจแบบ Dynamic (เว็บเพจแบบพลวัต) ได้ ซึ่งหมายถึงเว็บเพจที่มีเนื้อหาสาระและหน้าตาเปลี่ยนแปลงไปได้ในแต่ละครั้งที่ผู้ใช้เปิดดู โดยขึ้นอยู่กับเงื่อนไขต่าง ๆ เช่น ข้อมูลที่ผู้ใช้ส่งมาให้ หรือข้อมูลในฐานข้อมูล เป็นต้น

# โครงสร้าง HTML

HTML จะประกอบด้วยส่วนประกอบ 2 ส่วน ดังนี้

1. ส่วน Head คือส่วนที่จะเป็นหัว (Header) ของหน้าเอกสารทั่วไป หรือส่วนชื่อเรื่อง (Title) ของหน้าต่างการทำงานในระบบ Windows
2. ส่วน Body จะเป็นส่วนเนื้อหาของเอกสารนั้น ๆ ซึ่งจะประกอบด้วย Tag คำสั่งในการจัดรูปแบบหรือตกแต่งเอกสาร HTML



รูปที่ 5 โครงสร้าง HTML

- `<!DOCTYPE html>` กำหนด document นี้ เป็น HTML5
- `<html>` คือ root(อยู่ชั้นนอกสุด) element ของ page
- `<head>` ภายใน element นี้จะมีข้อมูลเกี่ยวกับ page
- `<body>` ภายใน element จะใช้แสดงเนื้อหาของ page

## <!DOCTYPE html>

การประกาศ <!DOCTYPE> เป็นการระบุชนิดของ document และช่วยให้ Browsers นั้นแสดงผลได้ถูกต้อง

```
<!DOCTYPE html>
<html>
<body>

<h1>My First Heading</h1>
<p>My first paragraph.</p>

</body>
</html>
```

รูปที่ 6 ตัวอย่างการประกาศ <!DOCTYPE html>

## <HTML>.....</HTML>

คำสั่ง <HTML> เป็นคำสั่งเริ่มต้นในการเขียนโปรแกรมและคำสั่ง </HTML> เป็นการสิ้นสุดโปรแกรม HTML คำสั่งนี้จะไม่แสดงผลในโปรแกรมเว็บเบราว์เซอร์

ดูรูปข้างบนเอา รูปเดียวกัน ˘(\*°▽°\*)˘

## <BODY>.....</BODY>

Body Section เป็นส่วนเนื้อหาหลักของหน้าเว็บ ซึ่งการแสดงผลจะต้องใช้ Tag จำนวนมาก ขึ้นอยู่กับลักษณะของข้อมูล เช่น ข้อความ, รูปภาพ, เสียง, วิดีโอ หรือไฟล์ต่างๆ ส่วนเนื้อหาเอกสารเว็บ เป็นส่วนการทำงานหลักของหน้าเว็บ

ดูรูปข้างบนเอา รูปเดียวกัน ( ' .. ) / ( .. ' )

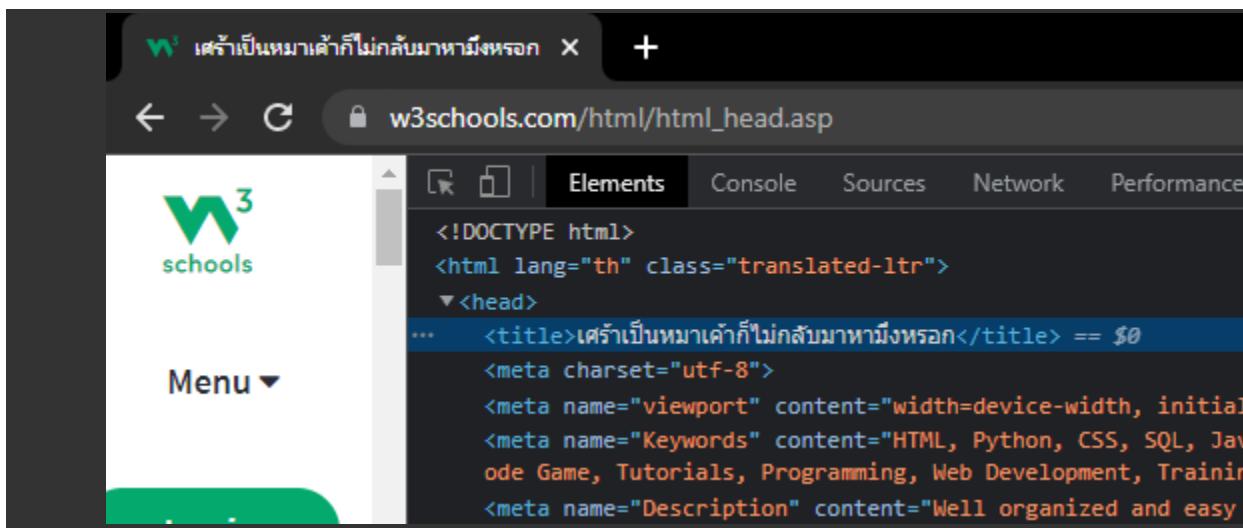
อ่าน อาจจะสองสัญ head หายไปไหน (พี่เจตนาอกให้เน้นเลยเขามาอันสุดท้าย)  
//จริงๆก็ไม่อยากทำหรอบหนึ่งอยแล้ว  
//แต่ก็เน้นๆละทำก็ได้

<HEAD>.....</HEAD>(ประชดแม่น)

Head Section เป็นส่วนที่ใช้อธิบายเกี่ยวกับข้อมูลเฉพาะของหน้าเว็บนั้น ๆ เช่น ชื่อเรื่องของหน้าเว็บ (Title), ชื่อผู้จัดทำเว็บ(Author) โดยส่วนของ head ยังเก็บhtmlส่วนที่จะลิงค์ หรือ โยงไปยังไฟล์ script ต่างๆอีกด้วย โดยจะมีรูปแบบต่างๆดังนี้

<title>

**<title>** เอาไว้แก้ชื่อข้อความกีประกูบัน title ตามชื่อ



รูปที่ 7 ตัวอย่างการใช้ <title>

## <style>

<style> เօາໄວ້ເຂົ້ານ CSS ແດ່ນີ້ແລະ

```
<style>
  body {background-color: powderblue;}
  h1 {color: red;}
  p {color: blue;}
</style>
```

ຮູບກີ່ 8 ຕົວຢ່າງການໃໝ່ <style>

## <link>

<link> ເօາໄວ້ລິງຄໍ CSS ໃນ HTML

```
<link rel="stylesheet" href="mystyle.css">
```

ຮູບກີ່ 9 ຕົວຢ່າງການໃໝ່ <style>

## <script>

<script> ເօາໄວ້ເຂົ້ານ javascript ປຣຶລິງຄໍ javascript ໃນ HTML ແດ່ນີ້ແລະ

```
<script type="text/javascript">
//<![CDATA[
var i = 10;
if (i < 5) {
  // some code
}
//]]>
</script>
```

ຮູບກີ່ \_ ເຂົ້ານໂດ້ຈavascript ໃນ HTML

```
<script src="myscripts.js"></script>
```

รูปที่ 10 ลิงค์โค้ด javascript ใน html

## <meta>

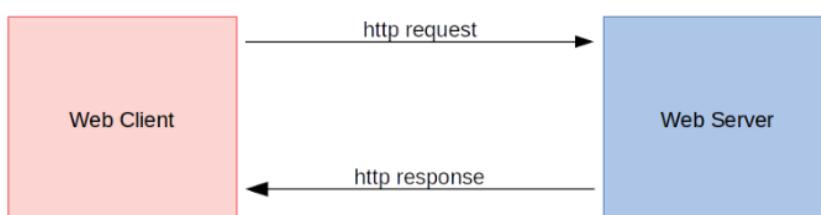
គឺជាការណែនាំ HTML ក៏ដូចបាយអ្នកប្រើប្រាស់វាដឹងទៅក្នុង Search Engine ដើម្បីដាក់ស្ថាបន្ទាល់ និងបង្ហាញអ្នកប្រើប្រាស់។ ការបង្ហាញអ្នកប្រើប្រាស់នេះមានរយៈពេលខ្លួន ដែលអាចបង្ហាញអ្នកប្រើប្រាស់បាន និងបង្ហាញអ្នកប្រើប្រាស់ដែលបានបង្ហាញ។

```
<meta charset="UTF-8">
<meta name="description" content="Free Web tutorials">
<meta name="keywords" content="HTML, CSS, JavaScript">
<meta name="author" content="John Doe">
```

### รูปที่ 11 ตัวอย่างการใช้ meta

## HTTP

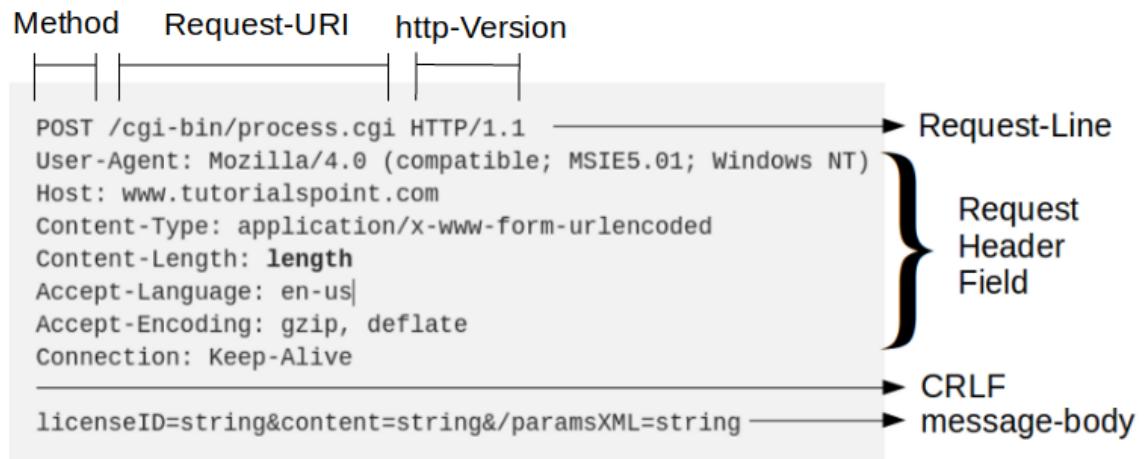
HTTP (Hypertext Transfer Protocol) គឺជាសម្រាប់ការផ្តល់ព័ត៌មានសម្រាប់ការប្រើប្រាស់ទូរទស្សន៍ និងការប្រើប្រាស់បច្ចុប្បន្ន។ ការប្រើប្រាស់ HTTP ត្រូវបានធ្វើឡើងដោយកិច្ចការប្រើប្រាស់ការណែនាំ និងការប្រើប្រាស់ការណែនាំ ដែលត្រូវបានរាយការណ៍ឡើង។ ការប្រើប្រាស់ HTTP ត្រូវបានធ្វើឡើងដោយកិច្ចការប្រើប្រាស់ការណែនាំ និងការប្រើប្រាស់ការណែនាំ ដែលត្រូវបានរាយការណ៍ឡើង។



รูปที่ 5 การแลกเปลี่ยนข้อมูลระหว่าง Web server และ Web Server ด้วย HTTP Protocol

## HTTP request

เป็นรูปแบบของ request จาก Client ที่ต้องการจะเข้ามายัง Server เมื่อเราコードงงส้างของ Request Message ไปเกี่ยวกับข้อมูลตอนที่เราใช้งานจริง จะแบ่งแต่ละส่วนได้แบบภาพข้างล่างนี้



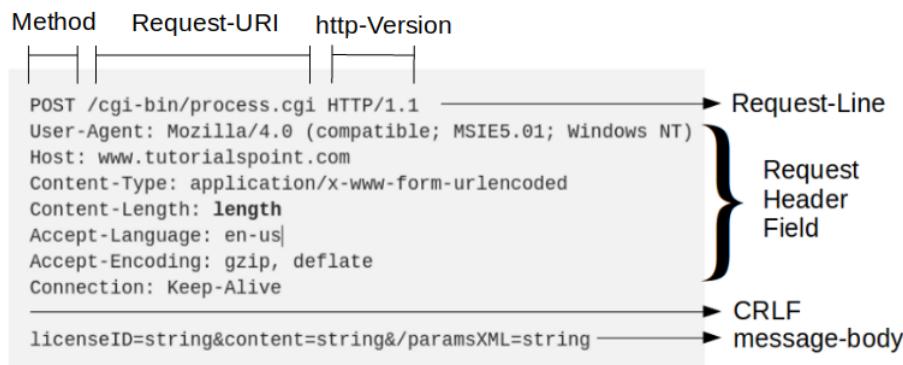
รูปที่ 12 ตัวอย่าง Request Message

```
Hypertext Transfer Protocol
> POST /adserver?u=97df6f8f08d8730261d4b44204353b4c&u=69832e95d26ae65e69ac72002a0be78c&z=50912&l=20121102085039&of=1.4&tm=15&g=1000002 HTTP/1.1\r\n
Host: ad.auditude.com\r\n
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; rv:16.0) Gecko/20100101 Firefox/16.0\r\n
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8\r\n
Accept-Language: en-US,en;q=0.5\r\n
Accept-Encoding: gzip, deflate\r\n
Connection: keep-alive\r\n
Referer: http://assets.newsinc.com/[[IMPORT]]/cdn.auditude.com/flash/modules/ndn-1.0/AuditudeAdUnit.swf\r\n
Content-type: application/x-www-form-urlencoded\r\n
> Content-length: 156\r\n
\r\n
[Full request URI: http://ad.auditude.com/adserver?u=97df6f8f08d8730261d4b44204353b4c&u=69832e95d26ae65e69ac72002a0be78c&z=50912&l=20121102085039&of=1.4&tm=15&g=1000002]
[HTTP request 2/10]
[Prev request in frame: 1960]
[Response in frame: 2148]
[Next request in frame: 2185]
File Data: 156 bytes
```

ภาพ Request กีก Capture จาก Wireshark

รูปที่ 7 Request Message กีก Capture โดย Wireshark

## ส่วนประกอบของ Request Message



รูปที่ 13 ตัวอย่าง Response Message

### - Request-Line

Request-Line จะเริ่มด้วย token method ตามด้วย Request-URI และ Version ของ Protocol จบด้วย CRLF (CRLF หมายถึง Carriage Return (\r) Line Feed (\n) หรือการขึ้นบรรทัดใหม่) ทุกส่วนจะถูกคั่นด้วยช่องว่าง (อักขระ SP) ตั้งภาพด้านบน

### - Request Method

HTTP Request จะกำหนด Action กีจะกระทำการต่อข้อมูล โดยที่ HTTP Method แต่ละตัว มีหน้าที่ที่ซัดเจนเพื่อตอบแแนวคิดการให้บริการ Resource ต่าง ๆ ให้กับ Client โดย วัตถุประสงค์ของ และ Method ดีอ

- GET ใช้เพื่อรับข้อมูลสถานะของ Resource นั้น ๆ
- HEAD ใช้เพื่อรับ Metadata ซึ่งเกี่ยวข้องกับ Resource นั้น ๆ
- PUT อาจจะใช้สำหรับการสร้าง Resource เข้าไปใน Resource ประเภท Store หรือ ปรับปรุงข้อมูลของ Resource

• DELETE ใช้ลบ Resource ออกจาก Store กีทำการเก็บข้อมูลไว้

- POST อาจจะใช้สร้าง Resource ใหม่เข้าไปใน Resource ประเภท Collection และใช้การสั่งให้ตัวควบคุม Resource ทำงานอื่น ๆ ที่ไม่ใช่ CRUD (Create — Read — Update — Delete)

โดยจะมีกฎเกณฑ์การใช้เพิ่มเติมดังนี้ **Rules of Request Methods**

- Request Header Fields

Header จะใช้สูญใช้สามารถส่งผ่านข้อมูลเพิ่มเติมเกี่ยวกับ

Request ของผู้ใช้ เช่น เป็นภาษาอะไร, ชนิดข้อมูลเป็นแบบไหน และนี่คือรายชื่อของ header กี่สำคัญ บางอย่างที่สามารถเลือกใช้ได้ เช่น

- Accept-Charset (ประเภทชุดตัวอักษรที่ยอมรับให้ใช้ เช่น Accept-Charset: UTF-8)
- Accept-Encoding (ประเภทการบีบอัดที่ยอมรับให้ใช้ เช่น Accept-Encoding: gzip)
- Accept-Language (ประเภทภาษาที่ยอมรับให้ใช้ เช่น Accept-Language: th)
- Connection (การเชื่อมต่อ ณ ตอนที่ส่ง Request เช่น Connection: keep-alive)
- Content-Length (ความยาวของเนื้อหา เช่น Content-Length: 221)
- Content-Type (ประเภทของเนื้อหา เช่น Content-Type: image/jpeg)
- Cookie จะกล่าวถึงในส่วนของ Cache/Cookies
- Host (ปลายทางที่จะส่ง request ไป)
- User-Agent (Web browser ที่ฝั่ง Client ใช้)

- Message-body

เป็นส่วนสุดท้ายของ Request message จะเป็นส่วนที่ใช้สำหรับส่งข้อมูลให้กับ Server อาจจะมีหรือไม่มีการส่งก็ได้ขึ้นอยู่กับ Method ที่เราเรียกด้วย อาทิเช่น POST Method หรือ HEAD Method อาจจะไม่มีในส่วนนี้ก็ได้เช่นกัน

## HTTP response

การ Response จะเกิดขึ้นหลังจากส่ง Request ไปที่ Server และ Server ก็จะมีการตอบรับกลับมาซึ่งเราจะเรียกว่าความที่ตอบกลับมาว่า HTTP-Response Message โดย Response Message จะประกอบไปด้วย 3 ส่วนหลักคือ Status-Line, Response Headers และ Message-body

```
Response = Status-Line
          *(( general-header
            | response-header
            | entity-header ) CRLF)
          CRLF
          [ message-body ]
```

รูปที่ 14 ตัวอย่าง Response Message

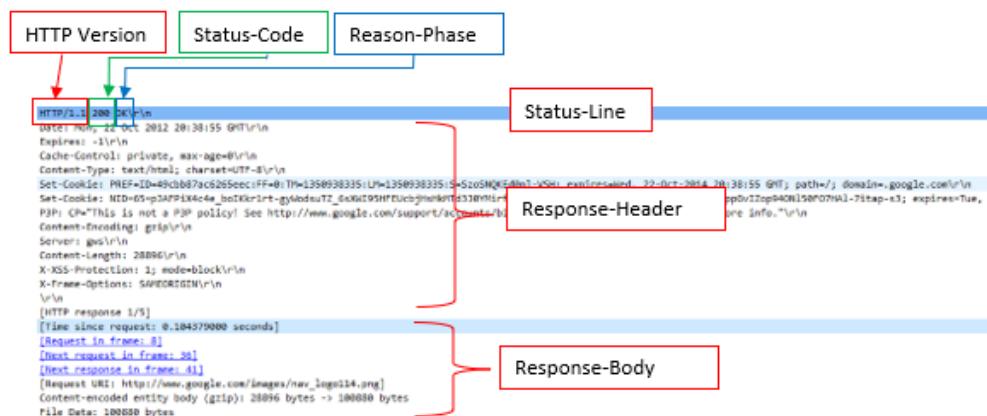
```

▼ Hypertext Transfer Protocol
  > HTTP/1.1 200 OK\r\n
    Date: Mon, 22 Oct 2012 20:38:55 GMT\r\n
    Expires: -1\r\n
    Cache-Control: private, max-age=0\r\n
    Content-Type: text/html; charset=UTF-8\r\n
    Set-Cookie: PREF=ID=49bb87ac6265eec:FF=0:TM=1350938335:LM=1350938335:S=SzoSNQKEd0n1-VSH; expires=Wed, 22-Oct-2014 20:38:55 GMT;
    Set-Cookie: NID=65=pJAFPiX4c4e_b0IKkr1rt-gyWodsutZ_6sXWl95HFEUcbjHshHkMTd330YMirf3_8gA9ZWl0W4pQrUPBaSSRJ4o82UEL0vv1kaXoppGvIZop94ON
    P3P: CP="This is not a P3P policy! See http://www.google.com/support/accounts/bin/answer.py?hl=en&answer=151657 for more info."\r
    Content-Encoding: gzip\r\n
    Server: gws\r\n
  > Content-Length: 28896\r\n
  X-XSS-Protection: 1; mode=block\r\n
  X-Frame-Options: SAMEORIGIN\r\n
  \r\n
  [HTTP response 1/5]
  [Time since request: 0.104379000 seconds]
  [Request in frame: 8]
  [Next request in frame: 36]
  [Next response in frame: 41]
  [Request URI: http://www.google.com/images/nav_logo114.png]
  Content-encoded entity body (gzip): 28896 bytes -> 100880 bytes

```

รูปที่ 15 Response Message ที่ถูก Capture โดย Wireshark

## ส่วนประกอบของ Response Message



รูปที่ 16 ส่วนประกอบของ Response Message

### - Status-Line

Status-Line จะเริ่มตัวด้วย Version ของ Protocol ตามด้วย Status-Code และ Reason-Phase จบด้วย CRLF (Carriage Return (\r) Line Feed (\n) หรือการขึ้นบรรทัดใหม่) ทุกส่วนจะถูกคั่นด้วยช่องว่าง (อักขระ SP) ดังภาพด้านบน

- Status-Code และ Reason-Phase

HTTP Status-Code จะแบ่งออกเป็นหมวดหมู่ตามเลขที่อยู่ตัวหน้าสุด พร้อมกับ Reason-Phase ที่จะอธิบาย HTTP Status-Code โดยจะมีรายละเอียดที่ใช้บ่อยดังนี้

- 200 OK เป็นมาตรฐานของ HTTP Response นั้น Success สำหรับ GET, PUT หรือ POST
- 201 Create เป็น Response สำหรับข้อมูลใหม่ได้ถูกสร้างขึ้น ใช้สำหรับ POST
- 204 No Content เป็น Response สำหรับ Request ที่ดำเนินการ Success แต่ไม่ได้ Return ข้อมูลกลับ
  - 304 Not Modified (RFC 7232) เป็น status code ที่บอกว่า client ได้รับการ Response แล้วอยู่ใน Cache และไม่จำเป็นจะต้องส่งผ่านข้อมูลเดิมอีกรึเปล่า
    - 400 Bad Request Request ที่ส่งมาโดย Client นั้นไม่ถูกดำเนินการ และ Server ไม่เข้าใจว่า Request ต้องการต้องการอะไร
      - 401 Unauthorized (RFC 7235) Client ไม่ได้รับอนุญาตในการเข้าถึงข้อมูลและควรจะส่ง Credential(หนังสือรับรองหรือลิขสิทธิ์) มาพร้อม request
      - 403 Forbidden บ่งบอกว่า Request นั้นถูกต้องและ Client ได้รับการอนุญาต แต่ Client ไม่ได้รับการอนุญาตให้เข้าถึงข้อมูลตัวบทุกอย่างประจำ
  - 404 Not Found บ่งบอกว่า Resource ที่ Request มาต้อง ยังไม่สามารถใช้งานได้ในขณะนี้ และคลื่นลมิกซ์เข้าถึงข้อมูลนี้ได
    - 410 Gone กรณีที่ร้องขอต้องไม่มีอยู่แล้ว ซึ่งอาจจะคล้ายคลึงกับ Status Code 404 เพียงแต่ว่าใน Status Code 410 นี้จะหมายถึง กรณีที่ร้องขอต้องของอาจจะถูกย้ายหรือถูกลบไปอย่างถาวร
      - 502 Bad Gateway ข้อผิดพลาด 502 Bad Gateway หมายความว่าเว็บเซิร์ฟเวอร์ที่เชื่อมต่อทำหน้าที่เป็นพื้นที่สำหรับการส่งต่อข้อมูลจากเซิร์ฟเวอร์อื่น แต่ได้รับการตอบสนอง (Response) ที่ไม่ดีจากเซิร์ฟเวอร์อื่นนั้น
      - 503 Service Unavailable บอกว่า Server ใช้งานไม่ได้ หรือไม่ว่างที่จะรับและดำเนินการ Request โดยส่วนใหญ่แล้ว server อยู่ในช่วงบำรุงรักษา

สำหรับ Status Code เพิ่มเติมนั้นสามารถอ่านต่อได้ที่นี่

- Response Header Fields

ตรงส่วนนี้จะแตกต่างกันออกไป โดยจะขึ้นอยู่กับ Request ที่ได้ทำการร้องขอและจำนวนผลลัพธ์ต่าง ๆ ที่เกิดขึ้นจากการประมวลผล จึงทำให้ผลของ Response Header Field ไม่จำเป็นจะต้องเหมือนกันในแต่ละ Packet

- Message-body

เป็นส่วนสุดท้ายของ Response message จะเป็นส่วนที่ใช้ส่งข้อมูลให้กับclient เอง อาจจะมีหรือไม่มีก็ได้ขึ้นอยู่กับ Method ที่ได้รับมาจาก Request อาทิเช่นถ้าเป็น HEAD Method อาจจะทำให้ไม่ต้องส่งข้อมูลในส่วนนี้ก็ได้ เช่นกัน

## Catch/Cookies

หมายถึง กลุ่มของข้อมูลที่ถูกส่งจากเว็บเซิร์ฟเวอร์ (Web Server) Majority เว็บเบราว์เซอร์ (Web browser) และถูกส่งกลับมายังเว็บเซิร์ฟเวอร์ทุก ๆ ครั้งที่เว็บเบราว์เซอร์ต้องขอข้อมูล โดยปกติแล้ว คุณก็จะถูกใช้เพื่อจัดเก็บข้อมูลขนาดเล็กๆ ไว้ที่เว็บเบราว์เซอร์ เพื่อให้เว็บเซิร์ฟเวอร์สามารถจดจำสถานะ การใช้งานของเว็บเบราว์เซอร์ที่มีต่อเว็บเซิร์ฟเวอร์

การใช้งานคุกกี้

คุกกี้ถูกนำไปใช้งานหลาย ๆ อย่าง ยกตัวอย่างเช่น

- จะจำข้อมูลของผู้ใช้เมื่อผู้ใช้กด save ข้อมูลผ่านหน้าเว็บไซต์
- จะจำเว็บไซต์ที่ผู้ใช้เคยเข้าในอดีต ทำให้สามารถย้อนดูประวัติการเข้าชมได้ในเมนู

ประวัติการเข้าชม ของเว็บбраузอร์นั้น ๆ

- เก็บข้อมูลตระกร้าลินค์ของผู้ใช้ในเว็บไซต์ e-commerce ที่ผู้ใช้ได้เคยหยิบสินค้าไว้ในตะกร้า
- แสดงโฆษณาที่ตรงเป้าหมายตามพฤติกรรมการท่องเว็บของผู้ใช้
- ตรวจสอบรายละเอียดการเข้าสู่ระบบของผู้ใช้

## HTTP With SSL

SSL (Secure Socket Layer) ซึ่งปัจจุบันได้พัฒนาขึ้นมาเป็น TLS (Transport Layer Security) คือ เทคโนโลยีในการเข้ารหัสข้อมูลสำหรับเพิ่มความปลอดภัยในการสื่อสารหรือส่งข้อมูลบนเครือข่ายอินเทอร์เน็ตระหว่างเครื่องเซิร์ฟเวอร์กับเว็บเบราว์เซอร์หรือ Application ที่ใช้งาน เพื่อให้ข้อมูลของผู้ใช้ปลอดภัยจากการเข้าถึงข้อมูลจากแฮกเกอร์ โดยวิธีการเรียกว่า จดจำผ่านโปรโตคอล HTTPS หรือ โปรโตคอลความปลอดภัยอื่น ๆ ตามแต่วิธีการใช้งาน

● วิธีการส่งข้อมูลแบบปกติ



● วิธีการส่งข้อมูลแบบ SSL



ประเภทของเทคโนโลยี SSL certificate

รูปที่ 17 การเปรียบเทียบกันระหว่าง HTTP กับ HTTPS

หลักการทำงานของ SSL

1. Client ขอเชื่อมต่อ กับ Server ผ่านโปรโตคอล SSL (URL นำหน้าด้วย https://)
2. Server ส่ง Certificate ที่ใช้ในการยืนยันตัวตน กลับมาให้ Client
3. Client สร้าง Key สำหรับเข้ารหัส และส่งกลับไปให้ Server
4. Server ใช้ Key ที่ได้รับ สร้างช่องทางการการเชื่อมต่อ ที่ปลอดภัย (Secure Session)



รูปที่ 18 ตัวอย่างการทำงานของ SSL

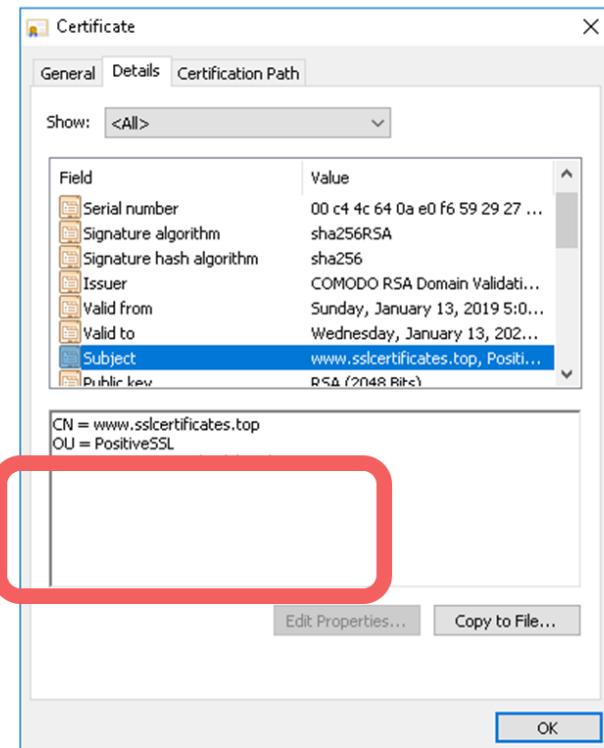
#### SSL certificate

SSL certificate คือ ใบรับรองอิเล็กทรอนิกส์ เป็นไฟล์ข้อมูลขนาดเล็กที่ได้มีการยูนิกไว้กับ Private Key ของเครื่องเซิร์ฟเวอร์ เพื่อยืนยันตัวตนและความถูกต้องในการส่งข้อมูลระหว่างเครื่องเซิร์ฟเวอร์กับเว็บเบราว์เซอร์หรือ Application ที่ใช้งาน มีการเข้ารหัสและถอดรหัสผ่านเทคโนโลยี SSL/TLS หากข้อมูลของผู้ใช้ถูกถักลับไปได้ ข้อมูลก็ผู้ใช้ก็ยังมีความปลอดภัย เพราะหากเกอร์จะไม่สามารถถอดรหัสข้อมูลของผู้ใช้ได้ เนื่องจากข้อมูลที่ได้ไป จะอยู่ในรูปแบบกีกูเข้ารหัส จะต้องมีคีย์ถอดรหัสที่เหมาะสมและตรงกันเท่านั้น ถึงจะสามารถถอดรหัสได้

## ประเภทของ SSL Certificate

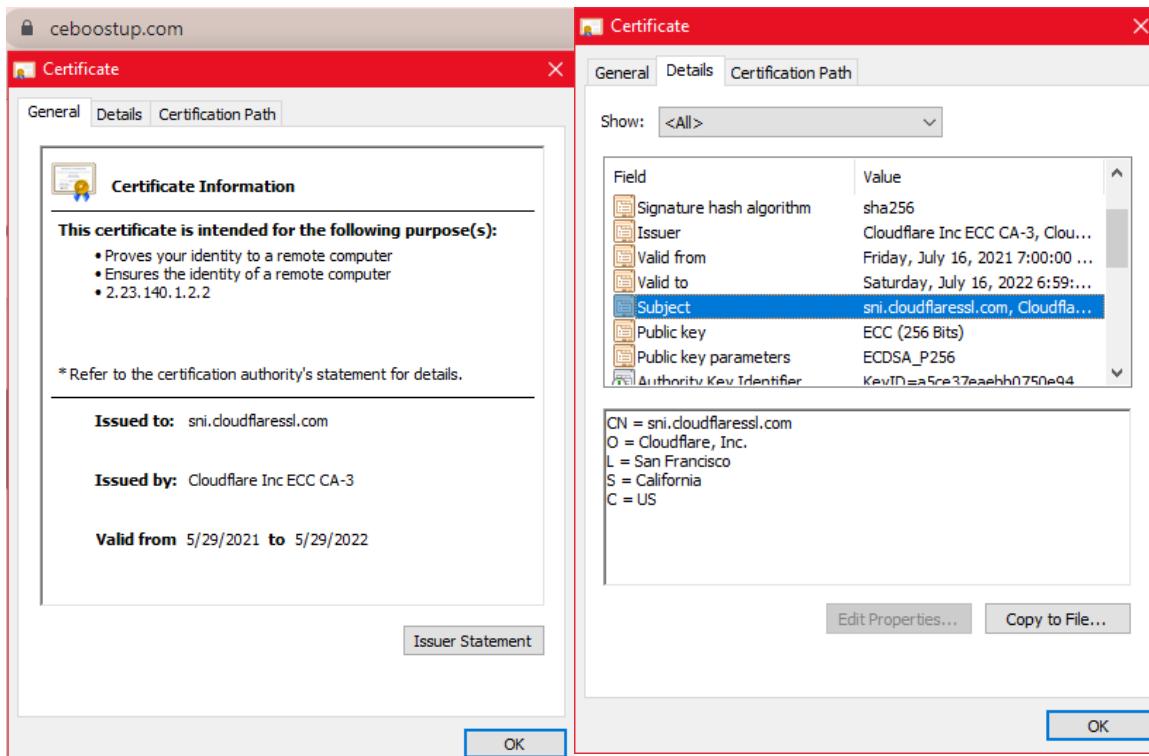
### 1. Domain Validation SSL Certificate (DV)

DV เป็นการออกใบรับรอง SSL ที่ง่ายและไวที่สุด โดยฝ่ายผู้ให้บริการ SSL จะมีการตรวจสอบเพียงว่าได้รับเงื่อนไขใดๆ ไม่ได้ตรวจสอบเชิงลึกแต่อย่างใด สามารถทำได้ในเวลาไม่กี่นาที หากเป็นเว็บไซต์ที่ใช้ใบรับรอง Domain Validation จะมีเพียงลักษณะคุณลักษณะเดียวและคำว่า Secure เก่านั้น



รูปที่ 19 Domain Validation SSL Certificate

### 2. Organization Validation SSL Certificate (OV)

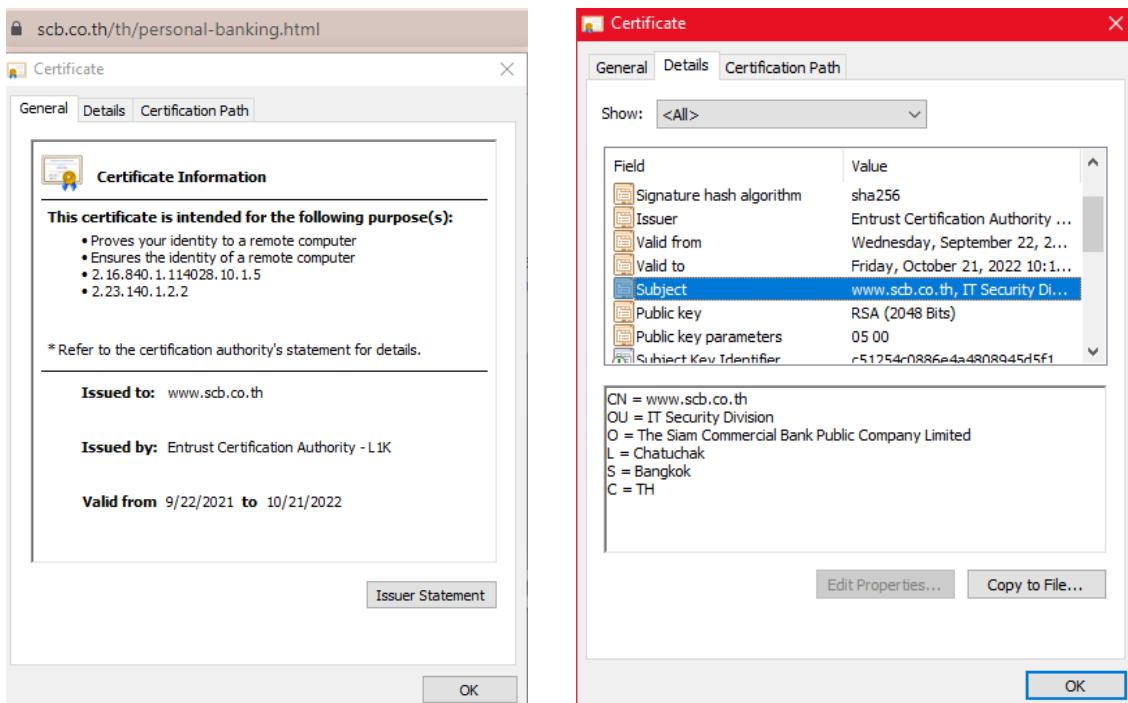


รูปที่ 20 Organization Validation SSL Certificate

### 3. Extended Validation SSL Certificate (EV)

EV เป็นการตรวจสอบที่เข้มงวดที่สุด สำหรับ Extended Validation ซึ่งนอกจากจะถูกตรวจสอบเจ้าของโดเมน ตรวจสอบองค์กรผ่านกรรมธุรกิจการค้าแบบเดียวกับ Organization Validation แล้ว ยังมีการตรวจสอบข้อมูลเชิงลึกของบริษัทด้วย เช่น มีการตั้งบริษัทมานานเท่าไหร่ เชื่อถือได้มากแค่ไหน โดยจะกินเวลาตรวจสอบราว ๆ หนึ่งถึงสองสัปดาห์

หากได้รับ Extended Validation และ ในช่อง URL จะขึ้นกุญแจสีเขียวพร้อมกับชื่อบริษัทมาให้ เราสามารถพบได้ในเว็บไซต์ของธนาคารหรือองค์กรในภาย ฯ ทั้งนี้ทั้งนั้นการตรวจสอบในรูปแบบต่าง ๆ อาจแตกต่างกันออกไปได้ตามผู้ให้บริการ SSL



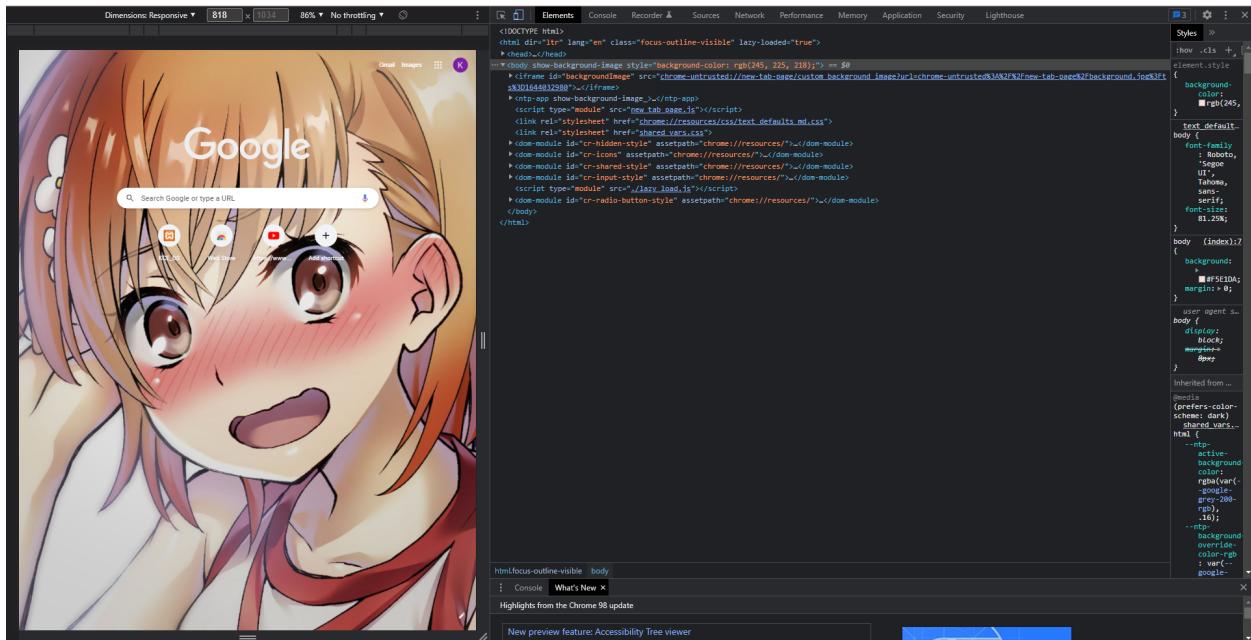
รูปที่ 21 Extended Validation SSL Certificate

# Inspector 101

## Overview

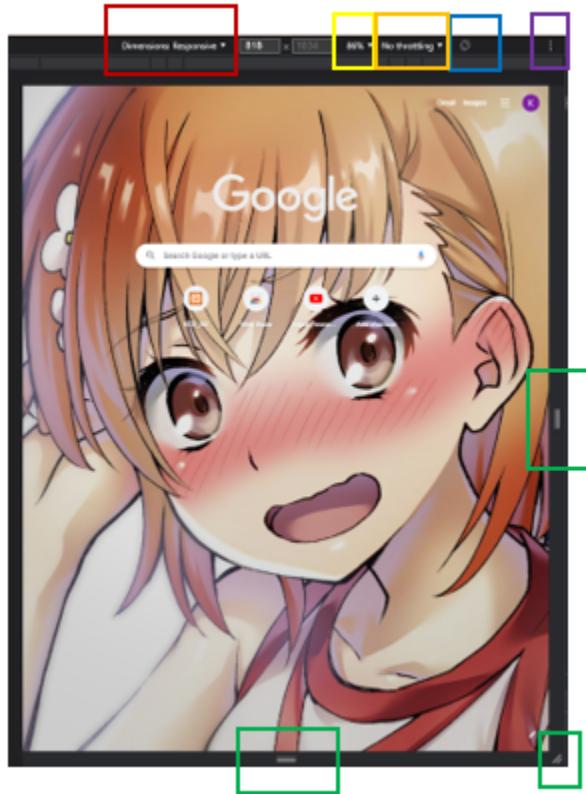
Inspector (Debugger หรือ Developer Tools) เป็นเครื่องมือที่ช่วยให้สามารถดูและแก้ไข element ต่าง ๆ บนหน้าเว็บและดูผลลัพธ์ได้กันเกี่ย (แต่จะเป็นการแก้ไขข้อความที่อยู่ใน DOM ไม่ส่งผลในการส่ง Request หรือ Response ใด ๆ ) รวมทั้งใช้ตรวจสอบโค้ดที่ผิดพลาดต่าง ๆ การดูค่าของตัวแปรและวัตถุของ JavaScript และอื่น ๆ ถ้าที่จะกล่าวต่อไปหลังจากนี้ โดย Inspector ในแต่ละбраузอร์จะสามารถเปิดได้โดยใช้ปุ่มลัดเหล่านี้

- Opera 12 มาพร้อมกับ Opera Dragonfly เรียกใช้ได้โดยการกด Ctrl+Shift+I (หรือคลิกขวาแล้วเลือก Inspect Element)
- Firefox มาพร้อมกับ Inspector เรียกใช้ได้โดยการกด Ctrl+Shift+I (หรือคลิกขวาแล้วเลือก Inspect Element)
- Chrome มาพร้อมกับ Developer tools เรียกใช้ได้โดยการกด Ctrl+Shift+I (หรือคลิกขวาแล้วเลือก Inspect Element) หรือกด F12
- Internet Explorer (Microsoft Edge) มาพร้อมกับ Developer tools เรียกใช้โดยการกด F12 โดย Inspector ของแต่ละбраузอร์จะมีลักษณะการใช้งานที่คล้ายคลึงกันดังภาพด้านล่างนี้



รูปที่ 22 ตัวอย่าง Inspector

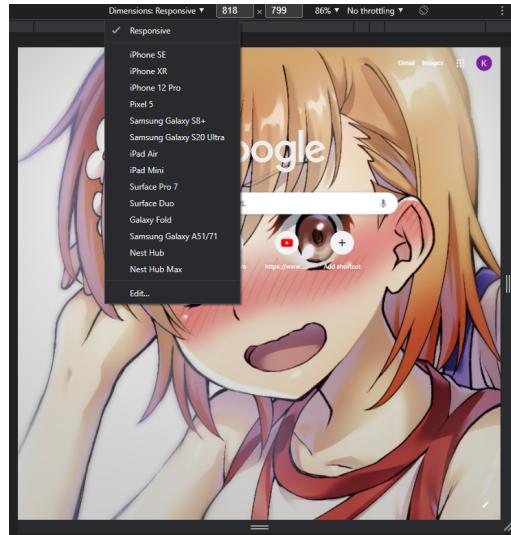
โดยแต่ละส่วนจะมีความสามารถดังนี้



รูปที่ 23 ส่วนประกอบของ Inspector ในผู้ช่วยการแสดงผล

#### Dimensional Response (กรอบสีแดง)

เป็นการปรับขนาดของหน้าจอให้มีขนาดต่าง ๆ โดยไม่ได้อ้างอิงกับหน้าจอที่ใช้อยู่อีกต่อไป ซึ่งจะสามารถเปลี่ยนให้เป็นขนาดหน้าจอและการแสดงผลบนหน้าจอของอุปกรณ์อื่น โดยขนาดและแสดงอยู่ที่ด้านขวาของกรอบสีแดงในรูปด้านบนเป็นขนาด กว้าง x สูง มีหน่วยเป็น px (หน่วยเดียวกับที่ใช้ใน CSS)



รูปที่ 24 การเลือกขนาดของหน้าจอให้มีขนาดเท่ากับอุปกรณ์อื่น ๆ

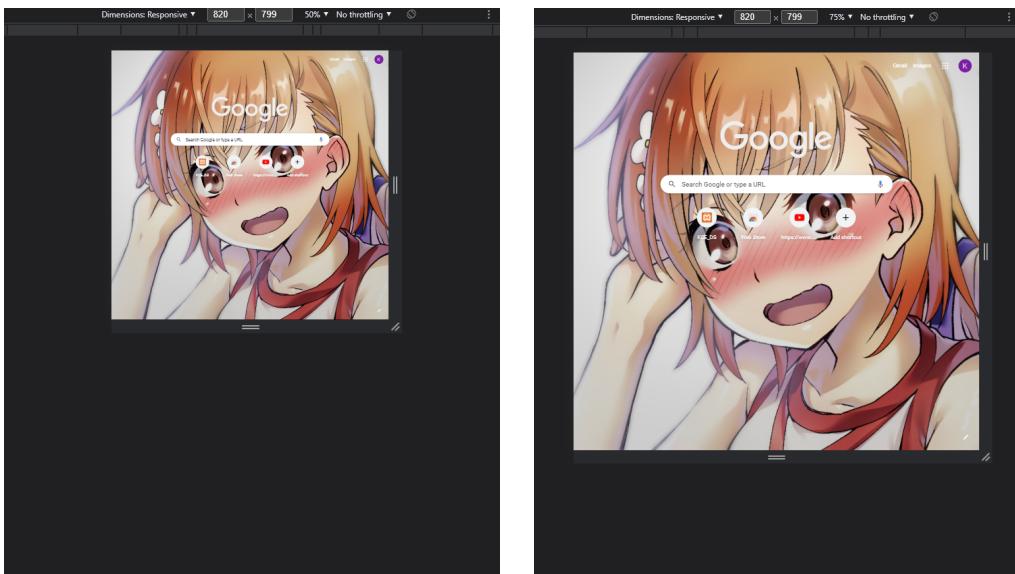
#### Independent Dimensional Response (กรอบสีเขียว)

เป็นการปรับขนาดของหน้าจอให้มีขนาดต่าง ๆ โดยไม่ได้อ้างอิงกับหน้าจอที่ใช้อยู่อีกต่อไป ซึ่งจะสามารถเปลี่ยนให้เป็นขนาดหน้าจอและการแสดงผลบนหน้าจอตามขนาดใดก็ได้โดยที่สามารถกำหนดได้สองวิธีดังนี้

- คลิกแล้วลากที่สัญลักษณ์ในกรอบสีเขียวจุดใดจุดหนึ่งในสามจุดนั้น
- ใส่ตัวเลขไปที่ส่วนการแสดงขนาดที่จะแสดงอยู่ที่ด้านขวาของกรอบสีแดงในรูปที่ 18

### Zoom (กรอบสีเหลือง)

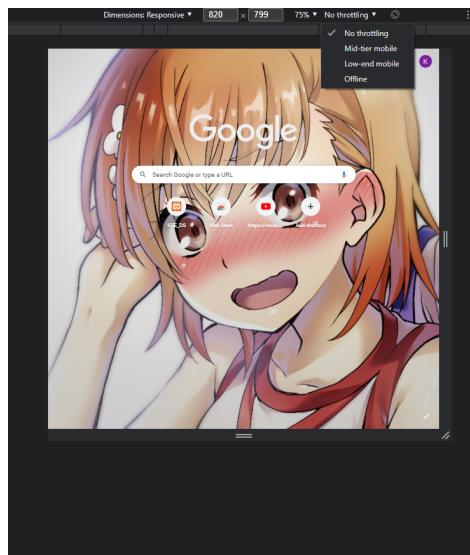
จะปรับการแสดงผลให้อยู่ในอัตราส่วนและขยายตามที่กำหนด



รูปที่ 25 ตัวอย่างการใช้ Zoom

### Throttling (กรอบสีส้ม)

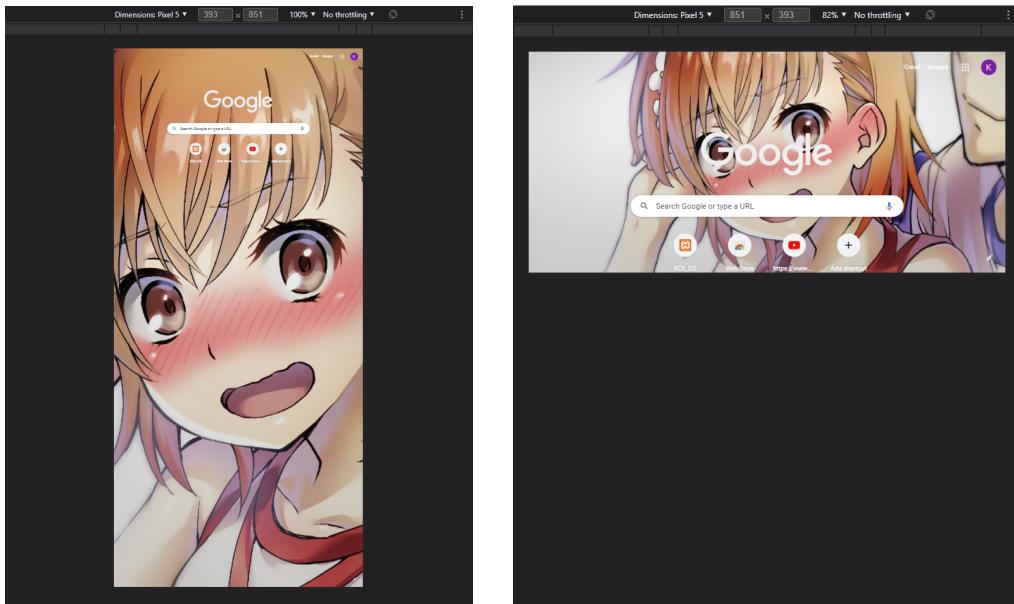
ใช้สำหรับจำลองเงื่อนไขความเร็วเครือข่ายต่าง ๆ ภายใต้อุปกรณ์ต่าง ๆ ซึ่งจะทำให้ในส่วนของ Network กับ Performance เปลี่ยนไป (สองอย่างนี้จะกล่าวถึงในส่วนต่อไป)



รูปที่ 26 ตัวเลือกของ Throttling

### Rotate (กรอบลีฟ้า)

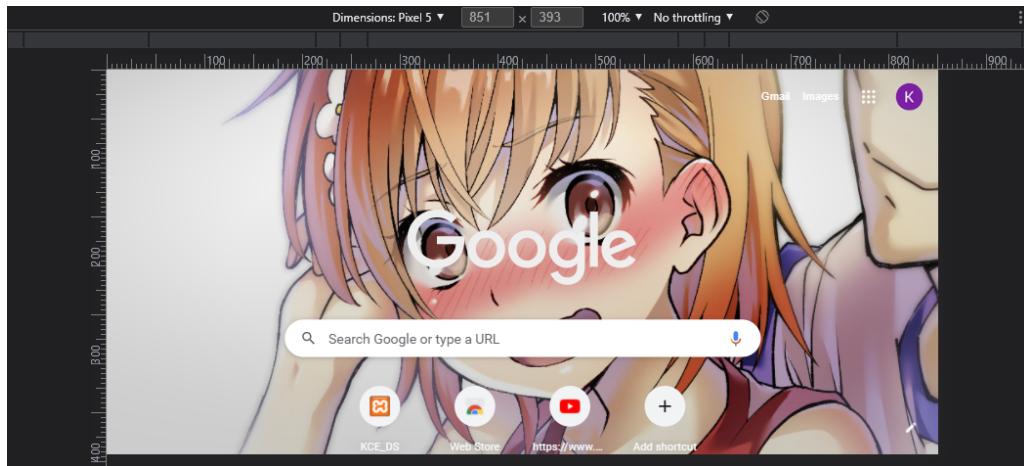
ใช้เพื่อหมุนด้วยการลับความกว้างและความสูง ณ ขณะนั้นกัน ด้วยการคลิกที่ลัญลักษณ์ในกรอบลีฟ้าในรูปที่ 18



รูปที่ 27 ตัวอย่างในการใช้ Rotate

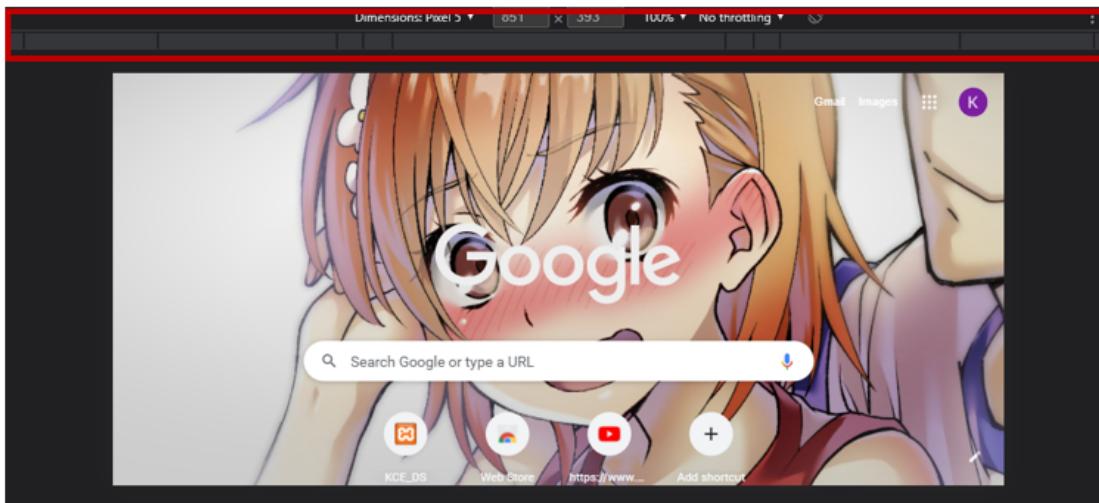
### More Option (กรอบลีฟอง)

เป็นตัวเลือกเพิ่มเติมในการปรับการแสดงผล เช่น Show Rulers (แสดงไม้บรรทัด)  
Reset to Default (ดีนค่ากลับไปสู่จุดเริ่มต้น) เป็นต้น



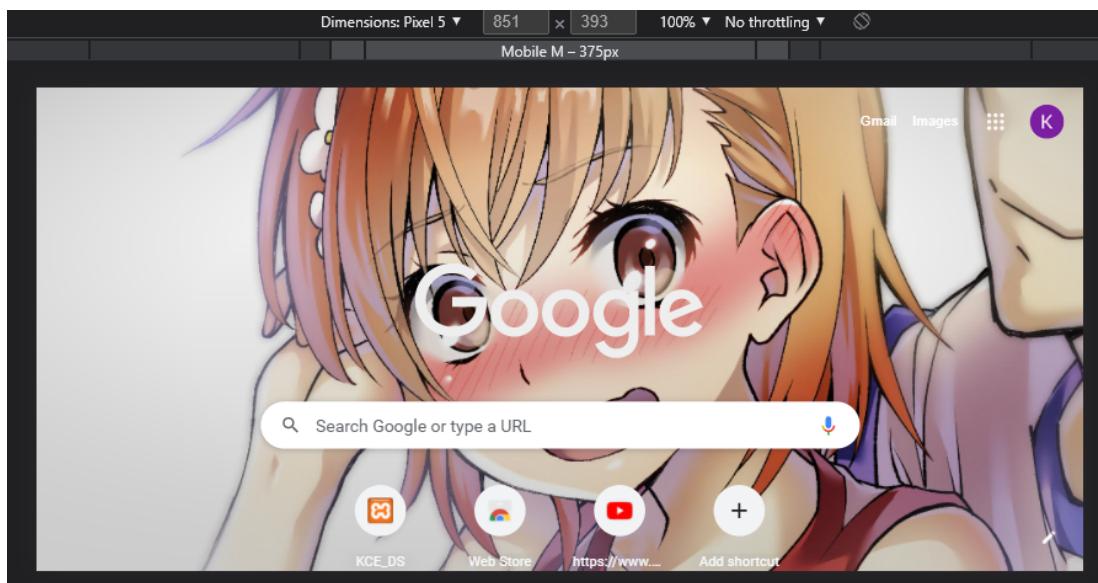
รูปที่ 28 ตัวอย่างที่ปรากฏใน Show Rulers

เพิ่มเติม



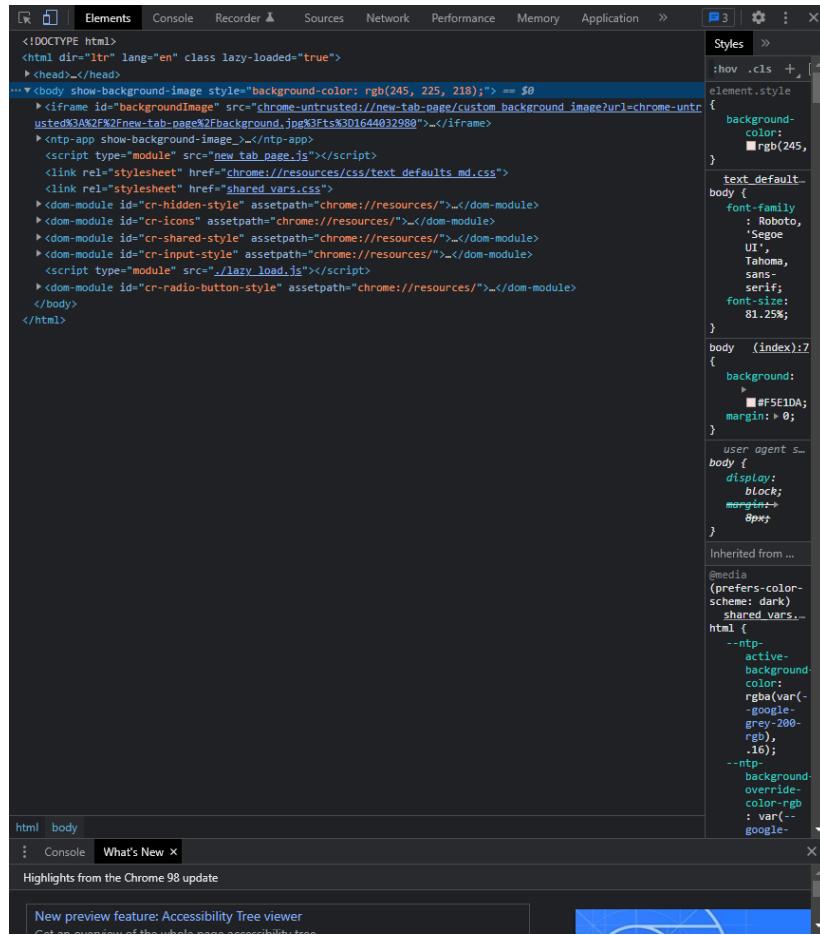
รูปที่ 29 แสดงความกว้างของอุปกรณ์

ภายในการออกแบบรูปด้านบนคือการแสดงความกว้างของอุปกรณ์มาตรฐานต่าง ๆ ซึ่งสามารถใช้ในการจัดการ media-queries ซึ่งสามารถดูได้ผ่านการลากเมาส์หรือเเชร์ฟ์ให้ลูกศรเอยู่เหนือแต่ละก้าว



รูปที่ 30 กรณีที่ต้องการรูปในความกว้างของหน้าจอโทรศัพท์มือถือ

## Toolbar

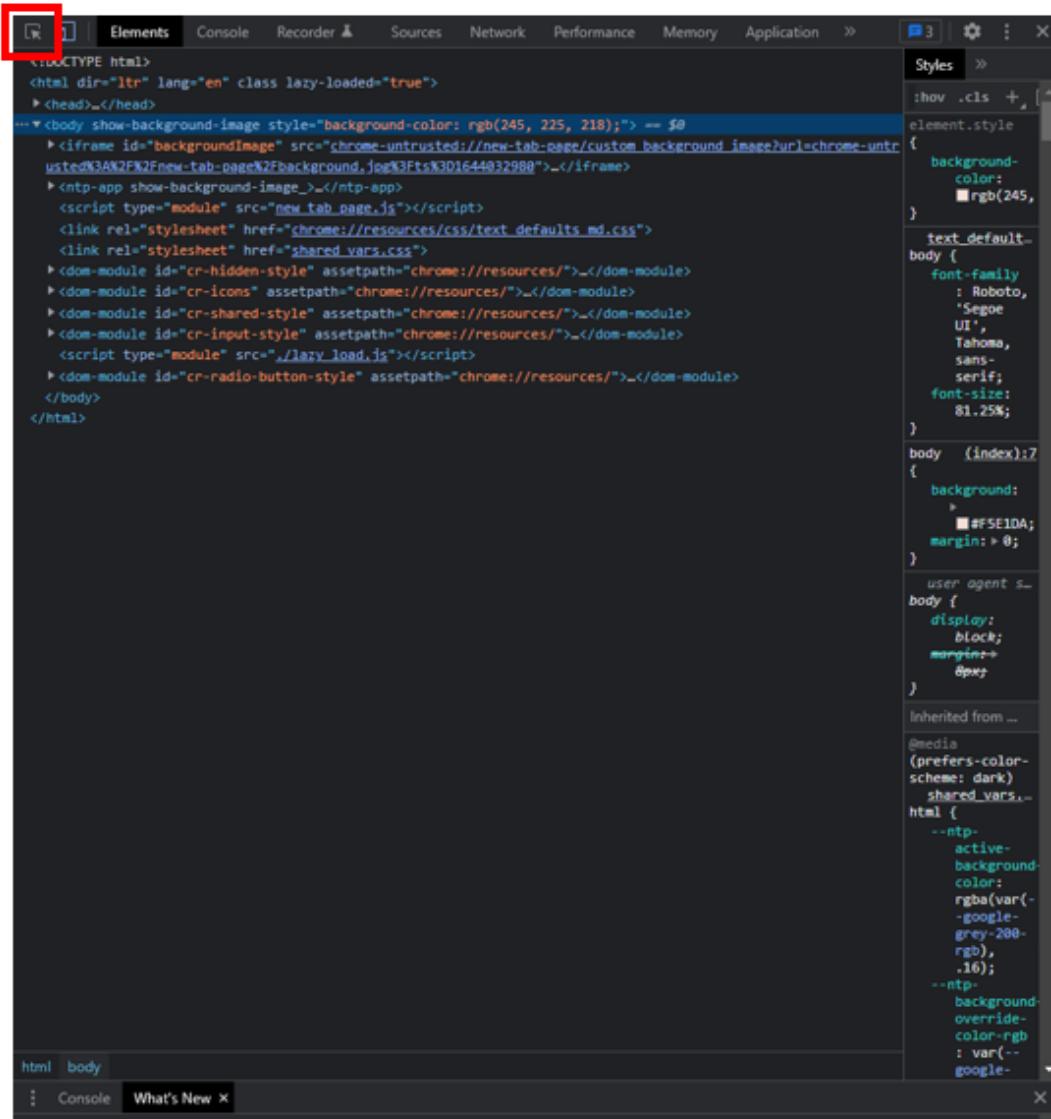


รูปที่ 31 ตัวอย่าง Toolbar ของ Inspector

จะเป็นแบบเครื่องมือสำหรับใช้ในการวิเคราะห์หน้าเว็บที่ผู้ใช้ต้องการ  
แต่ละส่วนดังนี้

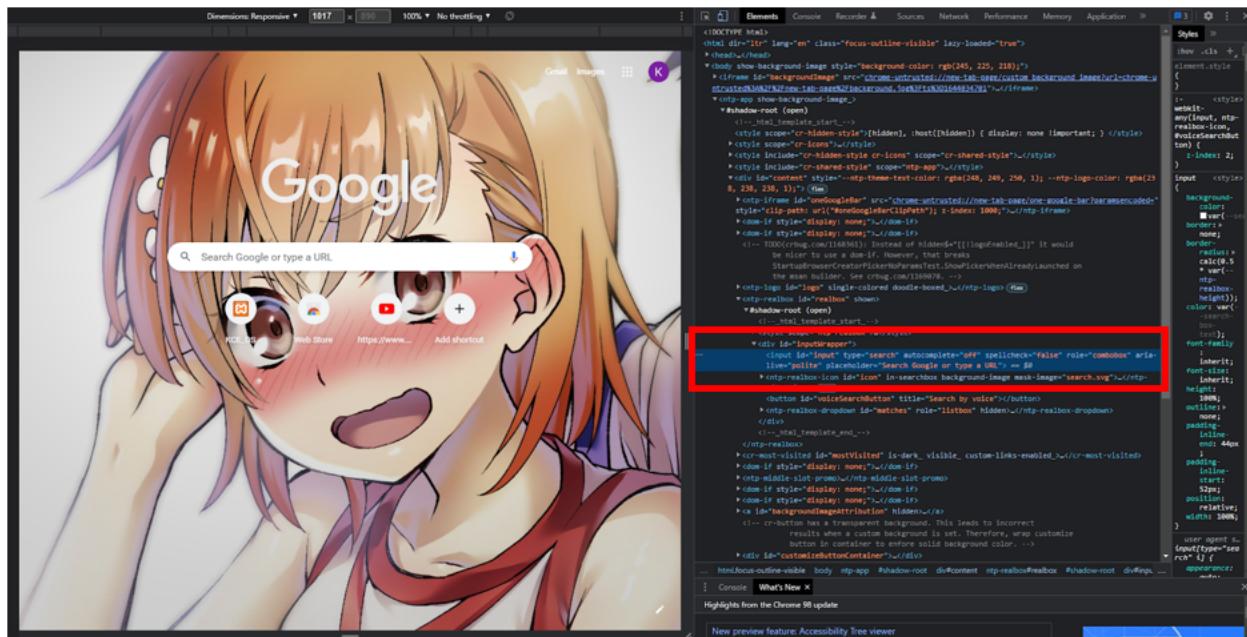
โดยจะมีรายละเอียดใน

## Select an element in the page to respect it



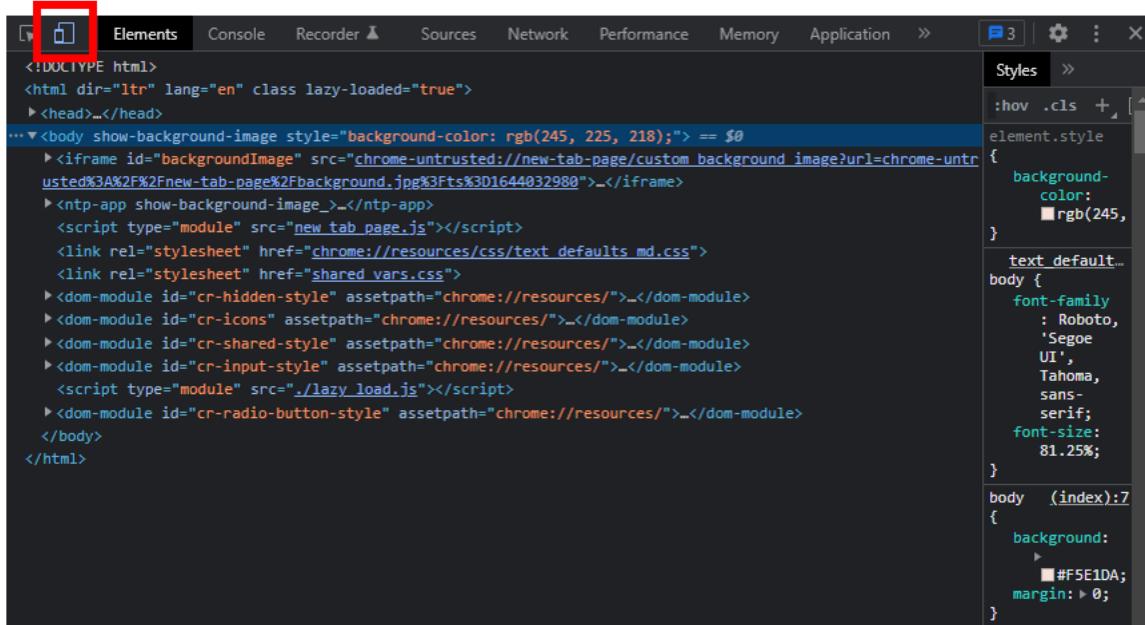
ຮູບກີ່ 32 ຕຳແໜ່ງຂອງ Select an element in the page to respect it

ສາມາຄຄລິກທີ່ສັນລັກບະລິນໃນກຣອບຮີອກດູ່ມືລັດ (Ctrl + Shift + C) ກີ່ໄດ້ ໂດຍກີ່ຈະໃຊ້ສໍາຮັບເລືອກສ່ວນທີ່ຈະຕ້ອງການວິເຄຣາທີ່ໄດ້ ເຊັ່ນ ເມື່ອຄລິກ Select an element in the page to respect it ແລ້ວ ເລືອກຄລິກໃນຫ່ອດັນຫາຂອງໜ້າເວັບໄຊຕໍ່ ຈະປາກງູເປັນຕຳແໜ່ງຂອງ HTML Code ຂອງສ່ວນດັ່ງກ່າວໃນແກບສິ້ນເງິນດ້ານຂວາມອື່ນກາພດ້ານລ່າງນີ້



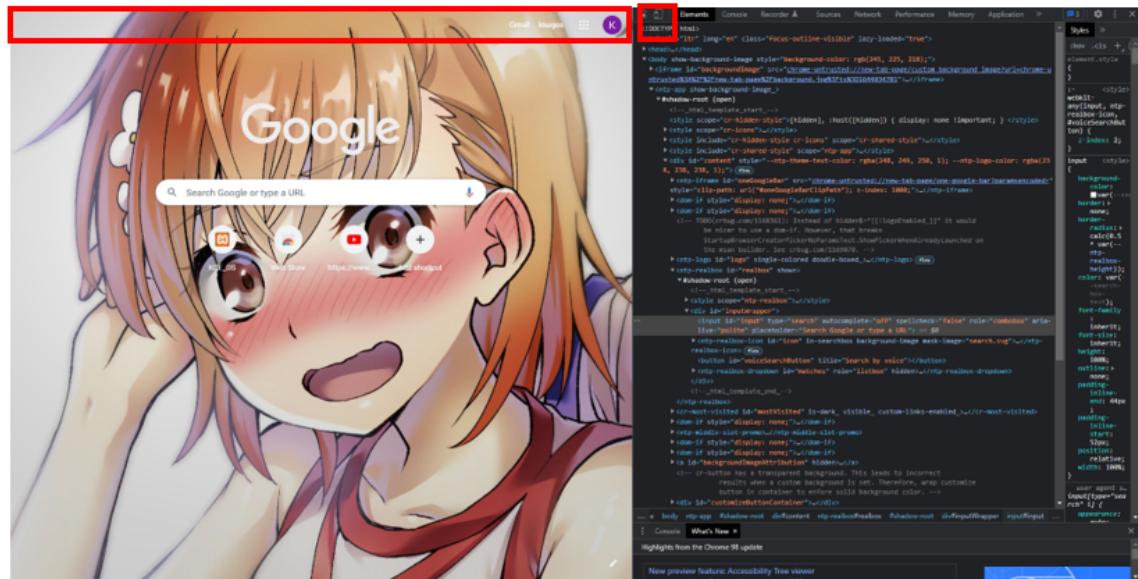
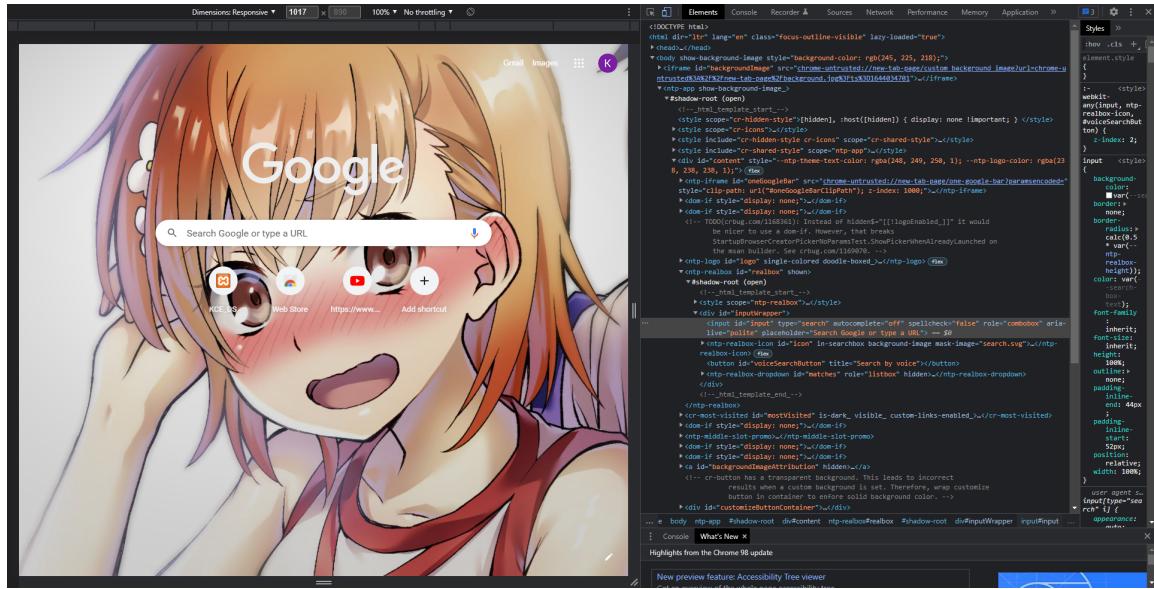
ຮູບກີ່ 33 ຕົວຢ່າງການໃຊ້ Select an element in the page to respect it

## Toggle device toolbar



ຮູບກີ່ 34 ຕຳແໜ່ນຂອງ Toggle device toolbar

สามารถคลิกที่สัญลักษณ์ในกรอบหรือกดปุ่มลัด (Ctrl + Shift + M) คือได้ โดยที่จะใช้สำหรับการเปิดและปิด Toggle device toolbar (เปลี่ยนการแสดงผลเป็นการแสดงผลในอุปกรณ์อื่น)



รูปที่ 35 การเปรียบเทียบเมื่อเปิด (บน) กับปิด (ล่าง) Toggle device toolbar

## Element

เป็นส่วนกี่แสดงถึง Code กี่ใช้ในการจัดวางในส่วนของ Element (สิ่งต่าง ๆ ที่ผู้ใช้มองเห็นบนหน้าเว็บ) ซึ่งจะรวมถึง Style (การตกแต่งหน้าเว็บด้วย CSS [Cascaded Style Sheet]) โดยที่เราสามารถแก้ไขเพื่อเปลี่ยนการแสดงผลได้โดยการเลือก Element และกดลิกกิที่หน้ากีแสดง Code (หน้าจอสีดำด้านขวาที่ถูกแบ่งเป็นสองส่วนคือ ส่วนแสดงผล (ด้านซ้าย) และส่วนกำหนดรูปแบบการแสดงผล (ด้านขวาที่อยู่ในแท็บ Style) ) แต่จะไม่ส่งผลกระทบต่อไฟล์หน้าเว็บ (ในบาง framework ไม่ได้เป็น HTML) ที่อยู่บนเซิร์ฟเวอร์ (เมื่อรีเฟรชหน้าเว็บจะกลับมาเป็นแบบเดิม)



รูปที่ 36 ตัวอย่างในการ Edit ที่ Element

## Toolbar յ່ອຍໃນແກ້ນບາງສ່ວນ

The screenshot shows the Chrome DevTools interface. On the left, the HTML panel displays the source code of a test page, which includes a noscript block, a root div, and various script tags. On the right, the Styles tab is active, showing a comprehensive list of CSS variables (SCSS) used by the application. These variables define colors, fonts, and other styling rules. The list is organized into sections like :root, --bs-blue, --bs-indigo, etc., with many specific color hex codes listed. The bottom of the DevTools window shows tabs for Console and What's New.

```
<!DOCTYPE html>
<html lang="en"> == $0
  > <head>...</head>
  > <body>
    ><noscript>You need to enable JavaScript to run this app.
    </noscript>
    ><div id="root">...</div>
    <!--
        This HTML file is a template.
        If you open it directly in the browser, you will see an empty page.

        You can add webfonts, meta tags, or analytics to this file.
        The build step will place the bundled scripts into the <body> tag.

        To begin the development, run `npm start` or `yarn start`.
        To create a production bundle, use `npm run build` or `yarn build`.
    -->
    <script src="/static/js/bundle.js" type="text/javascript">...</script>
    <script src="/static/js/vendors~main.chunk.js">...</script>
    <script src="/static/js/main.chunk.js">...</script>
  </body>
</html>
```

Styles Computed Layout Event Listeners DOM Breakpoints Properties Accessibility

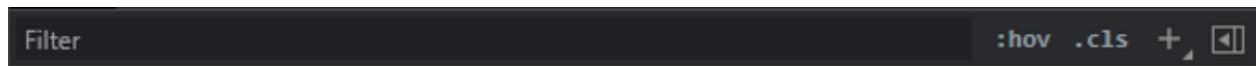
Filter :hov .cls + □

element.style { } @media (prefers-reduced-motion: no-preference) :root { scroll-behavior: smooth; } :root { --bs-blue: #0d6efd; --bs-indigo: #6610f2; --bs-purple: #6f42c1; --bs-pink: #e63984; --bs-red: #dc3545; --bs-orange: #fd7e14; --bs-yellow: #ffc107; --bs-green: #198754; --bs-teal: #20c997; --bs-cyan: #0dcaf0; --bs-white: #fff; --bs-gray: #6c757d; --bs-gray-dark: #343a40; --bs-gray-100: #f8f9fa; --bs-gray-200: #e9ecef; --bs-gray-300: #dee2e6; --bs-gray-400: #ced4da; --bs-gray-500: #adb5bd; --bs-gray-600: #6c757d; --bs-gray-700: #495057; --bs-gray-800: #343a40; --bs-gray-900: #212529; --bs-primary: #0d6efd; --bs-secondary: #6c757d; --bs-success: #198754; --bs-info: #0dcaf0; --bs-warning: #ffc107; --bs-danger: #dc3545; --bs-light: #f8f9fa; --bs-dark: #212529; --bs-primary-rgb: 13,110,253; --bs-secondary-rgb: 108,117,125; --bs-success-rgb: 25,135,84; --bs-info-rgb: 13,202,240; --bs-warning-rgb: 255,193,7; --bs-danger-rgb: 220,53,69; --bs-light-rgb: 248,249,250; --bs-dark-rgb: 33,37,41; --bs-white-rgb: 255,255,255; --bs-black-rgb: 0,0,0; --bs-body-color-rgb: 33,37,41; --bs-body-bg-rgb: 255,255,255; --bs-font-sans-serif: system-ui, -apple-system, "Segoe UI", Roboto, "Helvetica Neue", Arial, "Noto Sans", "Liberation Sans", sans-serif, "Apple Color Emoji", "Segoe UI Emoji", "Segoe UI Symbol", "Noto Color Emoji"; --bs-font-monospace: SFMono-Regular, Menlo, Monaco, Consolas, "Liberation Mono", "Courier New", monospace; --bs-gradient: linear-gradient(180deg, #rgba(255, 255, 255, 0.15),

ຮູບກີ່ 37 ຕົວຢ່າງໃນແກ້ນເພີ່ມເຕີມ

## Style

ເປັນສ່ວນຂອງ CSS Code ທີ່ໃຊ້ໃນການປະຈຸບັດຕ່າງໆການແສດງຜລ ໂດຍກີ່ຈະມີສ່ວນປະກອບດັ່ງນີ້



## Filter

เป็นส่วนที่ใช้ในการค้นหา CSS Selector กี่ผู้ใช้ได้ทำการระบุลงไป อาจจะค้นหาตาม State **:hov** หรือตาม CSS Class **.cls**

The screenshot shows the same UI as above, but with 'bs' selected in the search input. The results list includes properties like '--bs-primary' (hex #0d6efd), '--bs-secondary' (hex #6c757d), '--bs-success' (hex #198754), and so on, all with color swatches and hex codes. The 'Properties' tab is also visible at the top.

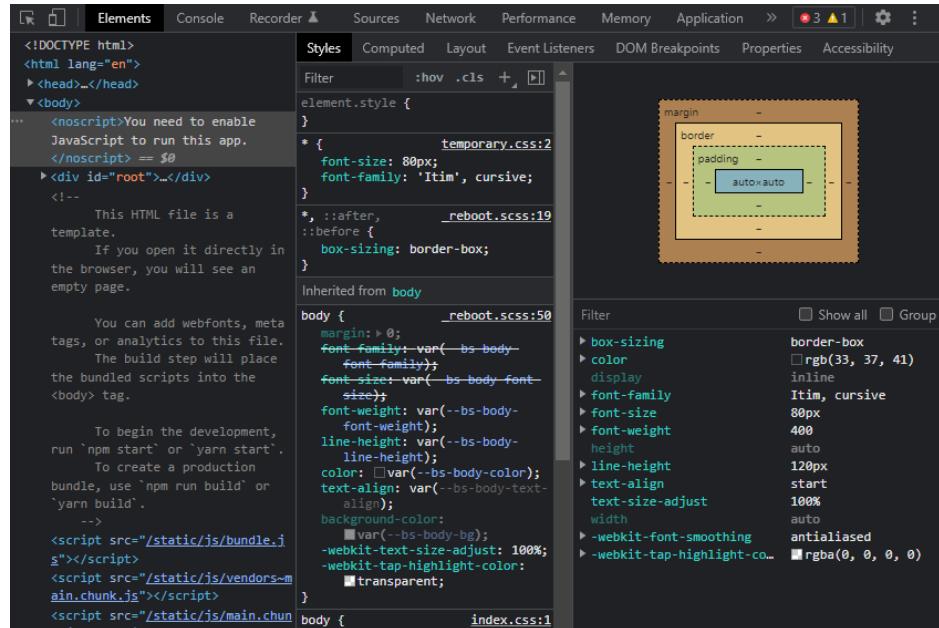
```
:root {  
  --bs-blue: #0d6efd;  
  --bs-indigo: #6610f2;  
  --bs-purple: #6f42c1;  
  --bs-pink: #d63384;  
  --bs-red: #dc3545;  
  --bs-orange: #fd7de14;  
  --bs-yellow: #ffc107;  
  --bs-green: #198754;  
  --bs-teal: #20c997;  
  --bs-cyan: #0dcaf0;  
  --bs-white: #fff;  
  --bs-gray: #6c757d;  
  --bs-gray-dark: #343a40;  
  --bs-gray-100: #f8f9fa;  
  --bs-gray-200: #e9ece1;  
  --bs-gray-300: #dee2e6;  
  --bs-gray-400: #ced4da;  
  --bs-gray-500: #adb5bd;  
  --bs-gray-600: #6c757d;  
  --bs-gray-700: #495057;  
  --bs-gray-800: #343a40;  
  --bs-gray-900: #212529;  
  --bs-primary: #0d6efd;  
  --bs-secondary: #6c757d;  
  --bs-success: #198754;  
  --bs-info: #0dcaf0;  
  --bs-warning: #ffc107;  
  --bs-danger: #dc3545;  
  --bs-light: #f8f9fa;  
  --bs-dark: #212529;  
  --bs-primary-rgb: 13,110,253;  
  --bs-secondary-rgb: 108,117,125;  
  --bs-success-rgb: 25,135,84;  
  --bs-info-rgb: 13,202,240;  
  --bs-warning-rgb: 255,193,7;  
  --bs-danger-rgb: 220,53,69;  
  --bs-light-rgb: 248,249,250;  
  --bs-dark-rgb: 33,37,41;  
  --bs-white-rgb: 255,255,255;  
  --bs-black-rgb: 0,0,0;  
  --bs-body-color-rgb: 33,37,41;  
  --bs-body-bg-rgb: 255,255,255;  
  --bs-font-sans-serif: system-ui,-apple-system,"Segoe UI",Roboto,"Helvetica  
  Neue",Arial,"Noto Sans","Liberation Sans",sans-serif,"Apple Color Emoji","Segoe UI  
  Emoji","Segoe UI Symbol","Noto Color Emoji";  
  --bs-font-monospace: SFMono-Regular,Menlo,Monaco,Consolas,"Liberation Mono","Courier  
  New",monospace;  
  --bs-gradient: linear-gradient(180deg, #rgba(255, 255, 255, 0.15),  
  #rgba(255, 255, 255, 0));  
  --bs-body-font-family: var(--bs-font-sans-serif);  
  --bs-body-font-size: 1rem;  
  --bs-body-font-weight: 400;  
  --bs-body-line-height: 1.5;  
  --bs-body-color: #212529;  
  --bs-body-bg: #fff;
```

## รูปที่ 39 ตัวอย่างการใช้ Filter

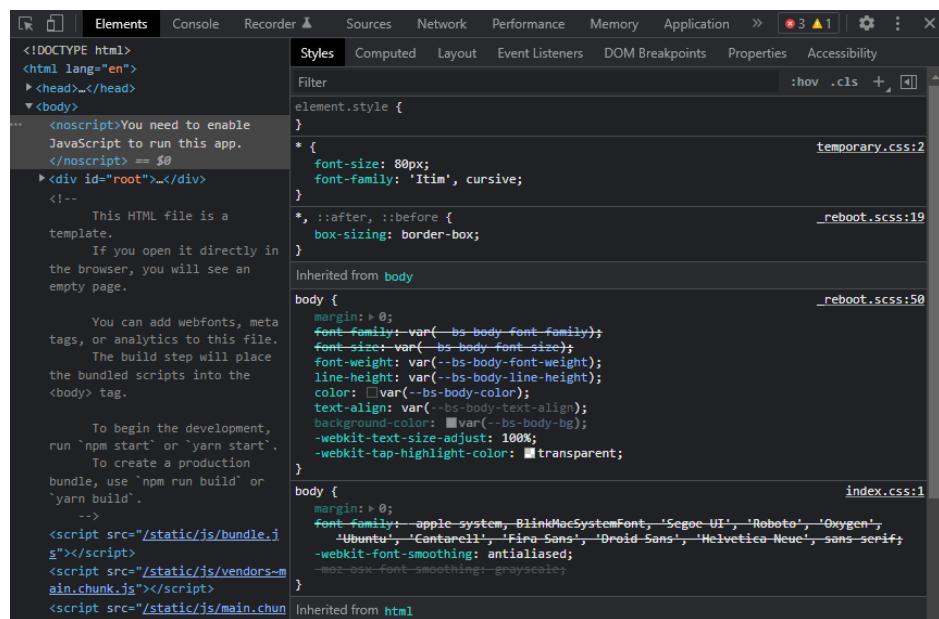
ตามรูปที่ 39 หากพิมพ์ว่า bs filter จะทำการกรองจนเหลือเพียง CSS Variable กี่ซึ่งว่า bs เพียงอย่างเดียวเท่านั้น

## Show computed style sidebar

ເປັນການນໍາຂໍ້ມູນ Computed ອອກມາແສດງກີ່ດ້ານຂວາງພື້ນທີ່ Style



The screenshot shows the Chrome DevTools interface with the 'Elements' tab selected. The left panel displays the HTML structure of a file named 'index.html'. The right panel shows the 'Computed' tab of the Styles panel, which lists the computed styles for the 'body' element. A large preview window on the right shows a box model diagram for the 'body' element, illustrating the relationship between margin, border, padding, and the auto-sized content area.

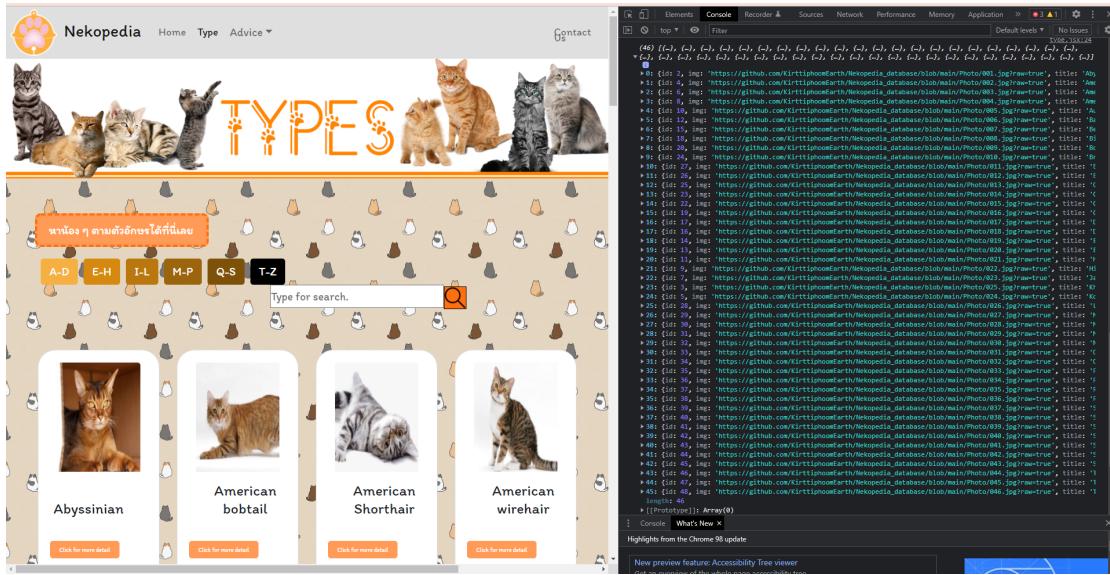


This screenshot shows the same setup as the previous one, but the computed styles listed in the sidebar are different. The 'body' element now has a 'font-family' of 'apple-system, BlinkMacSystemFont, 'Segoe UI', 'Roboto', 'Oxygen', 'Ubuntu', 'Cantarell', 'Fira Sans', 'Droid Sans', 'Helvetica Neue', sans-serif; and a '-moz-osx-font-smoothing' property set to 'grayscale'.

ຮູບກີ່ 40 ຕົວຢ່າງກ່ອນ (ບນ) ແລະ ລັ້ງ (ລ່າງ) ໃຊ້ Show computed style sidebar

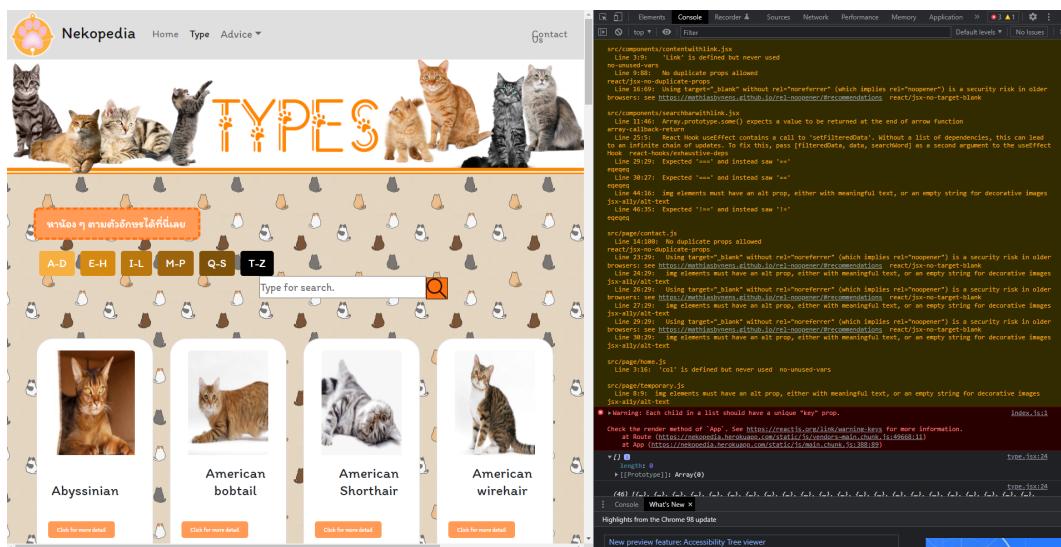
## Console

เป็นส่วนที่เป็นการแสดงผลกีไม่ใช่การแสดงผลทางหน้าเว็บโดยตรง ซึ่งการทำงานนั้นคล้ายกับ terminal ใน Visual Studio Code หรือ Command Prompt ของ Windows นั่นเอง



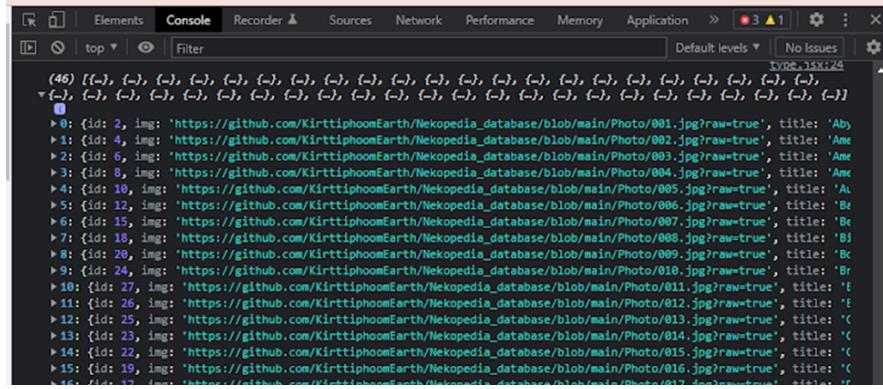
รูปที่ 41 ตัวอย่างการแสดงข้อมูลใน Console

ดังภาพ จะเห็นได้ว่าใน Website ข้างต้นนั้นได้ทำการ fetch ข้อมูลแล้วนำมาแสดงผล ที่ด้านหน้าเว็บ วิถีก็งั้นใช้และค่านั้นผ่าน Console ด้วย (คล้ายคลึงกับ printf ในภาษาซีนั่นเอง) ก็จะสามารถดูข้อมูลเหล่านี้ได้ผ่านส่วนของ Console ซึ่งนอกจากนี้ยังสามารถดู Warning หรือ Error ที่อาจเกิดขึ้นได้ผ่านหน้าเว็บเช่นกัน



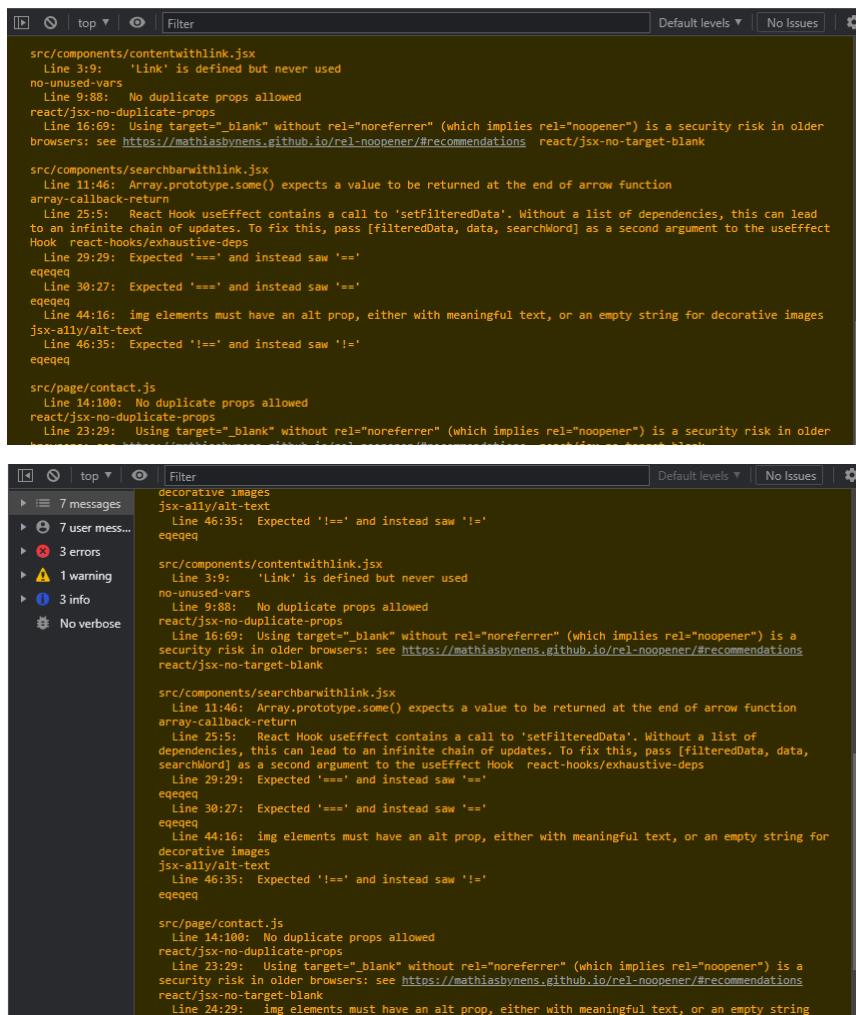
รูปที่ 42 ตัวอย่างการแสดง Warnings และ Errors ใน Console

## Toolbar ย่ออยู่ในแท็บบางส่วน



รูปที่ 43 ตัวอย่างของ Toolbar ย่ออยู่ในแท็บบางส่วนของ Console

## นำ Console sidebar มาแสดงด้านซ้ายของ Console



รูปที่ 44 ตัวอย่างก่อน (บน) และหลัง (ล่าง) การใช้ Console sidebar

## ⌚ ทำการล้างค่าข้อมูลใน Console ออกไปจากการแสดงผล

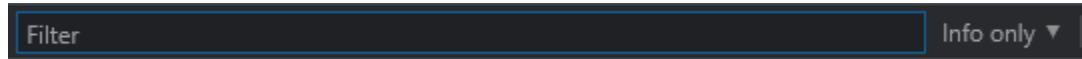
The screenshot displays the CodeLenser interface from the Microsoft Visual Studio Code extension. It consists of two vertically stacked code editor panes.

The top pane shows a file named `src/components/contentwithlink.jsx`. The code contains several ESLint errors, which are highlighted in yellow and provide detailed explanations:

- `Line 3:9: 'Link' is defined but never used`
- `no-unused-vars`
- `Line 9:88: No duplicate props allowed`
- `react/jsx-no-duplicate-props`
- `Line 16:69: Using target="_blank" without rel="noopener" (which implies rel="noopener") is a security risk in older browsers: see https://mathiasbynens.github.io/rel-noopener/#recommendations react/jsx-no-target-blank`
- `src/components/searchbarwithlink.jsx`
- `Line 11:46: Array.prototype.some() expects a value to be returned at the end of arrow function array-callback-return`
- `Line 25:5: React Hook useEffect contains a call to 'setFilteredData'. Without a list of dependencies, this can lead to an infinite chain of updates. To fix this, pass [filteredData, data, searchWord] as a second argument to the useEffect Hook react-hooks/exhaustive-deps`
- `Line 29:29: Expected '===' and instead saw '=='`
- `eqlreq`
- `Line 30:27: Expected '===' and instead saw '=='`
- `eqlreq`
- `Line 44:16: img elements must have an alt prop, either with meaningful text, or an empty string for decorative images jsx-ally/alt-text`
- `Line 46:35: Expected '!== and instead saw '!='`
- `eqlreq`
- `src/page/contact.js`
- `Line 14:100: No duplicate props allowed`
- `react/jsx-no-duplicate-props`
- `Line 23:29: Using target="_blank" without rel="noopener" (which implies rel="noopener") is a security risk in older browsers: see https://mathiasbynens.github.io/rel-noopener/#recommendations react/jsx-no-target-blank`

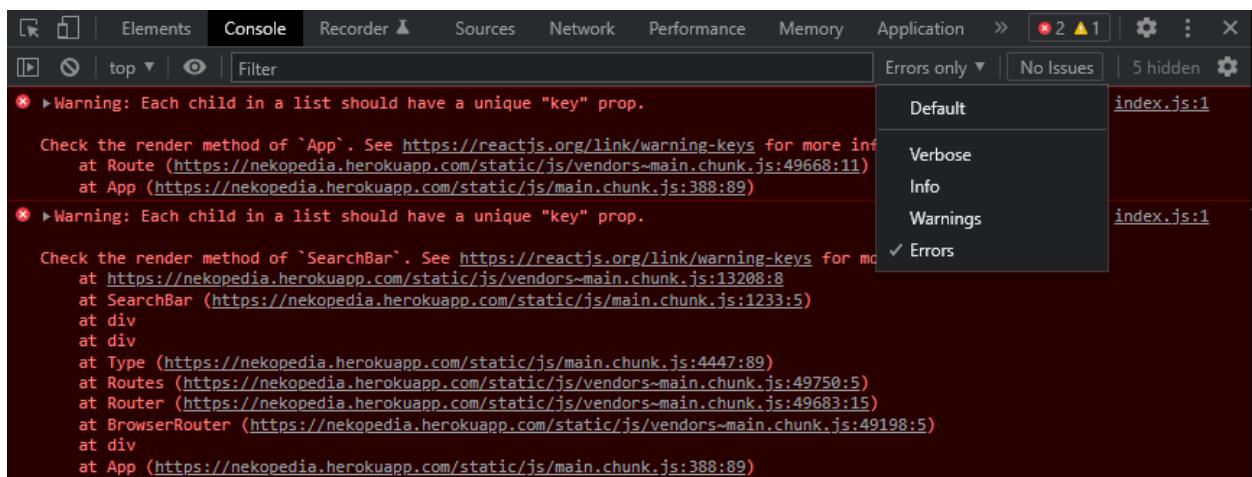
The bottom pane is also a code editor, but it is currently empty, showing only the navigation bar.

รูปที่ 45 ตัวอย่างก่อน (บน) และหลัง (ล่าง) การใช้ล้างค่าข้อมูลใน Console

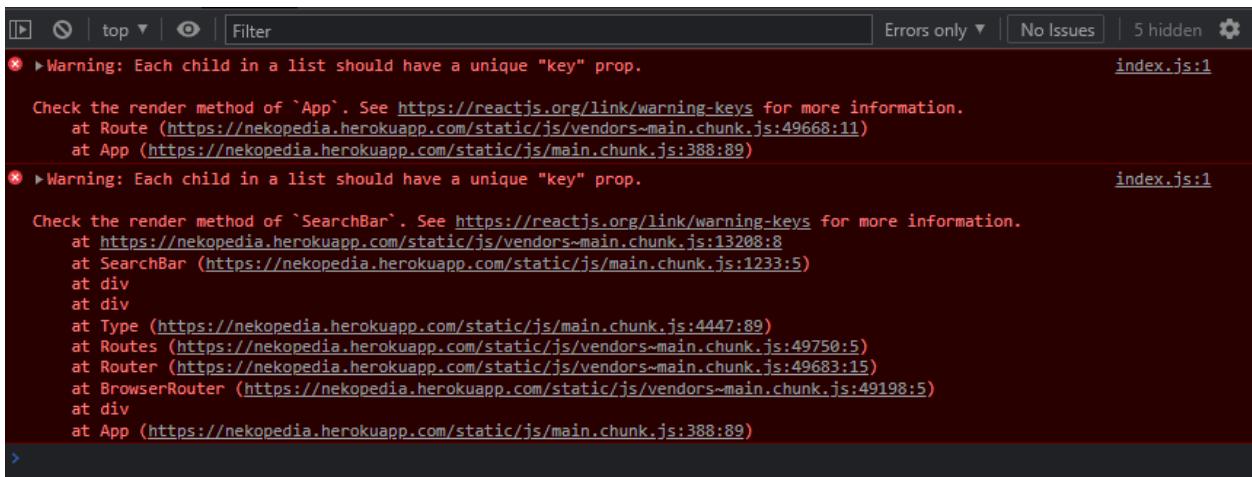


ຮູບກີ່ 46 ຈົດ Filter

ຕົວກຮອງຂໍ້ມູນໃນ Console ເຊັ່ນ ນຳຂໍ້ມູນລາຍ່າງເດືອຍ ຂຣື່ວເລືອກ Warning ພາຍ່າງເດືອຍ ຂຣື່ວເລືອກເຂົພາະ Warning ກັບ Error ມາ ເປັນຕົ້ນ



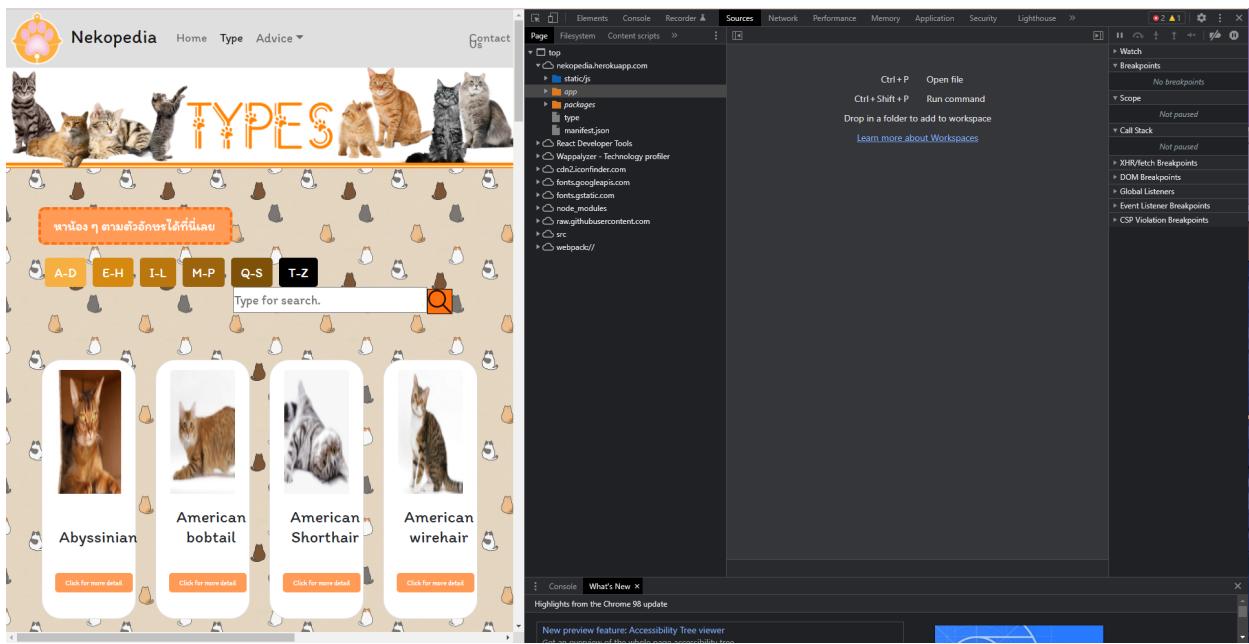
ຮູບກີ່ 47 ຕົວເລືອກໃນກາຮກຮອງປະເກົກຂໍ້ມູນ



ຮູບກີ່ 48 ຕົວຢ່າງໃນກາຮກຮອງປະເກົກຂໍ້ມູນ (ເລືອກເປັນກຮອງແຕ່ປະເກົກ Warning ອ່າງເດືອຍເກົ່າໜັ້ນ)

## Sources

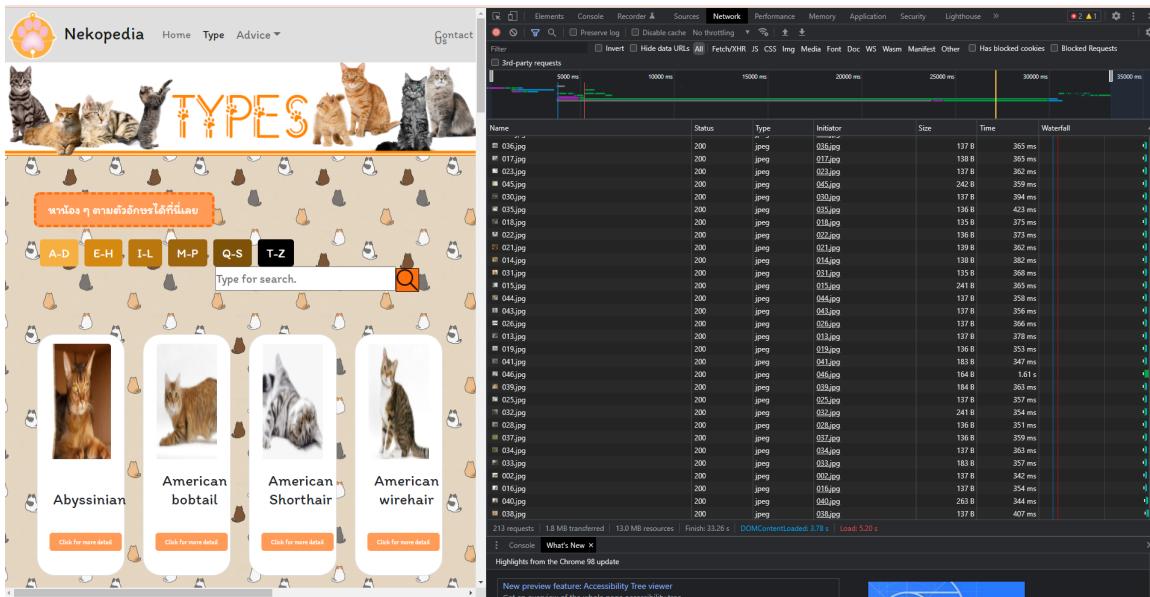
ເປັນສ່ວນກີ່ແສດງຄື່ງ Code ດລ້າຍກັບ Element ແຕ່ສ່ວນກີ່ແຕກຕ່າງគື້ມີໄຟໄລ໌ຈະແສດງແດ່ສ່ວນກີ່ອູ່ໜ້າເວັບອ່າຍາເດືອນນັ້ນ ຍັງຈະຮວມຄື່ງສົດຮີປົກທີ່ມີຢູ່ເບື້ອງໜັງ ເຊິ່ງໃຫ້ສ່ວນກີ່ໃຊ້ ແລະກັບພຍາກຮີທີ່ຈະຍັງມີຄຸກໂໂລດຂຶ້ນມາໃຊ້ຈານດ້ວຍ



ຮູບກີ່ 49 ຕັວຢ່າງໜ້າແກ້ບ Sources

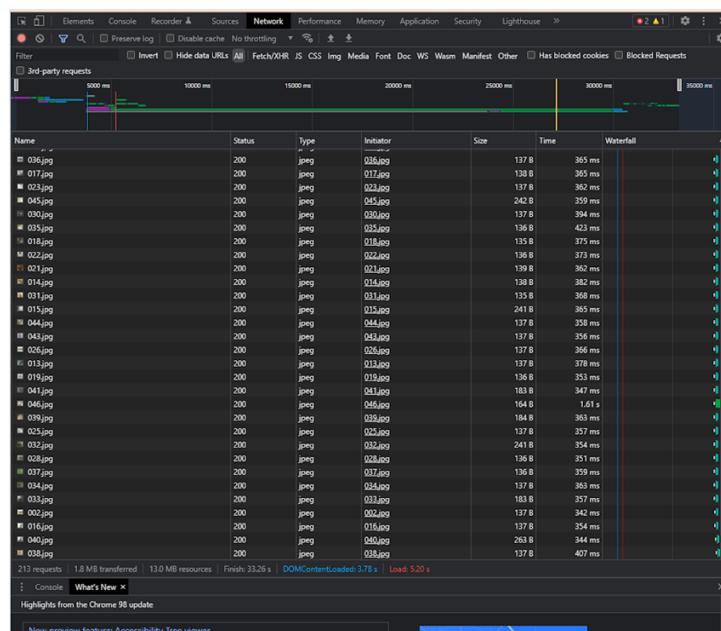
## Network

จะเป็นส่วนกี่ใช้สังเกตการเดลีอันกี่ของกรัพยากรผ่านเน็ตเวิร์กในช่วงเวลาใดเวลาหนึ่ง



รูปที่ 50 ตัวอย่างหน้าแท็บ Network

โดยที่จะมีส่วนประกอบดังต่อไปนี้



รูปที่ 51 ส่วนประกอบหน้าแท็บ Network

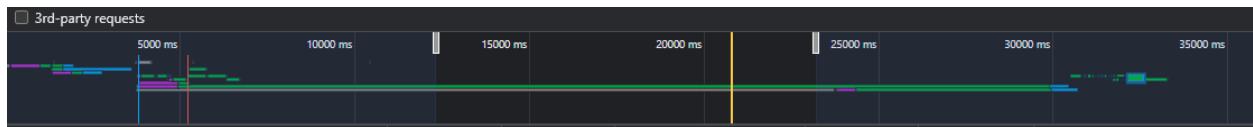
## Filter



รูปที่ 52 ช่อง Filter

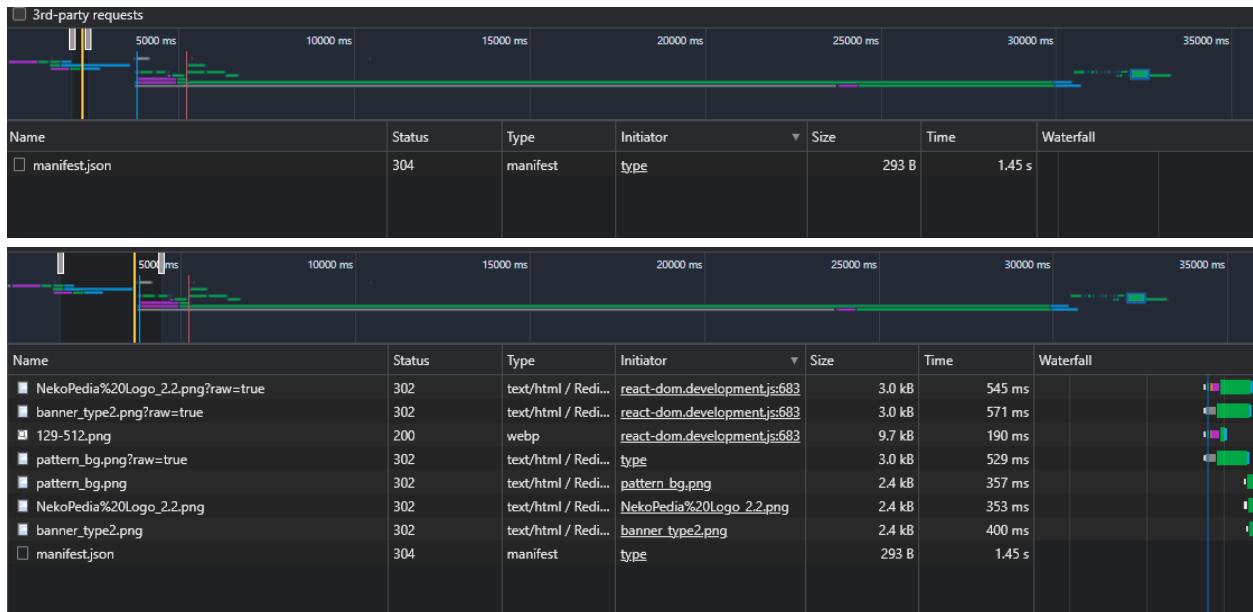
Filter เป็นส่วนที่ใช้ทำการกรองข้อมูลให้แสดงอุปกรณ์ที่ผู้ใช้ต้องการ อาทิเช่น เลือกอุปกรณ์รูปภาพ เลือกมาเฉพาะส่วนที่เป็นสคริปต์เท่านั้น เป็นต้น

## Timeline



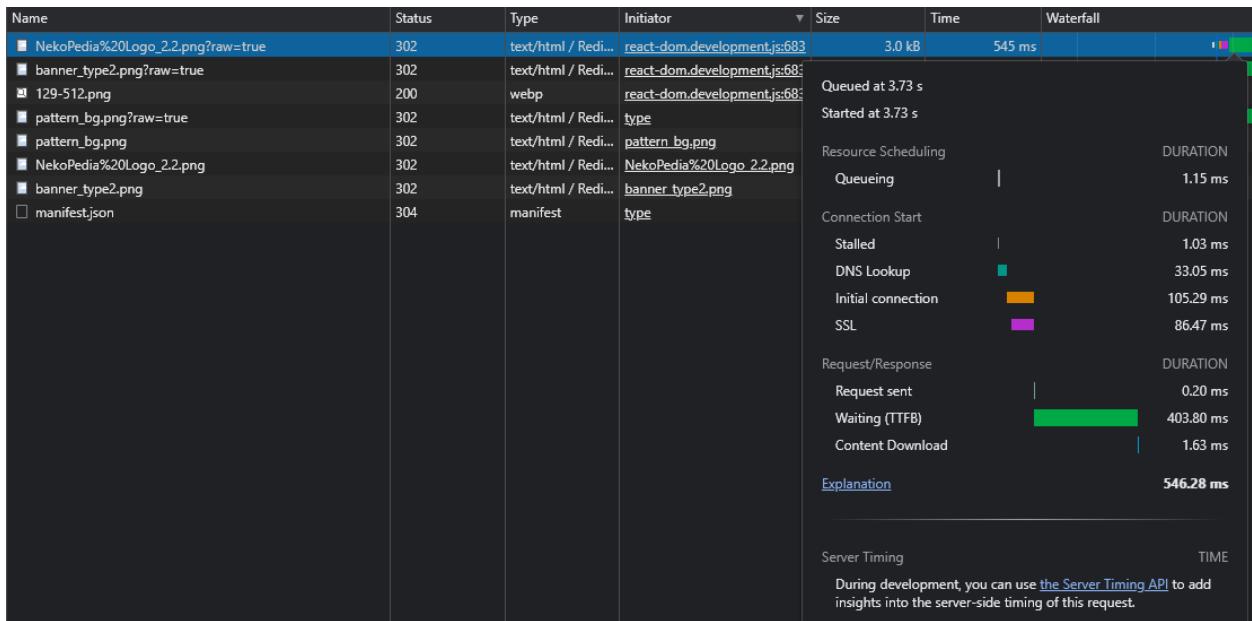
รูปที่ 53 ช่อง Timeline

เป็นส่วนที่จะแสดงการโหลดข้อมูลในช่วงเวลาเดียวกัน โดยเราสามารถคลิกเพื่อดู ณ จุดใดจุดหนึ่งหรือจะลากเพื่อดูเป็นช่วงเวลาใดเวลาหนึ่งก็ได้ เช่นกัน



รูปที่ 54 การเลือกสังเกตจาก Timeline ณ จุดใดจุดหนึ่ง (บน) หรือ ณ ช่วงเวลาใดช่วงเวลาหนึ่ง (ล่าง)

ซึ่งถ้าหากเลือกที่กรัพยากรอย่างใดอย่างหนึ่งนั้น จะสามารถดูรายละเอียด Status code ประเภทไฟล์ ขนาดไฟล์ เวลาที่ใช้ในการโหลดและ Waterfall (Timeline ของแต่ละ Process) ของกรัพยากรได้



รูปที่ 55 การเลือกสังเกตจากกรัพยากรอันໄດ້ອັນຫົ່ງ

## Application

เป็นส่วนที่สามารถดูส่วนของ Service ต่าง ๆ ที่ทำงานอยู่เบื้องหลังของเว็บไซต์ที่ทำการลังเกตอยู่ โดยจะรวมถึง Storage และ Cookies ที่ใช้ด้วยเซนกัน

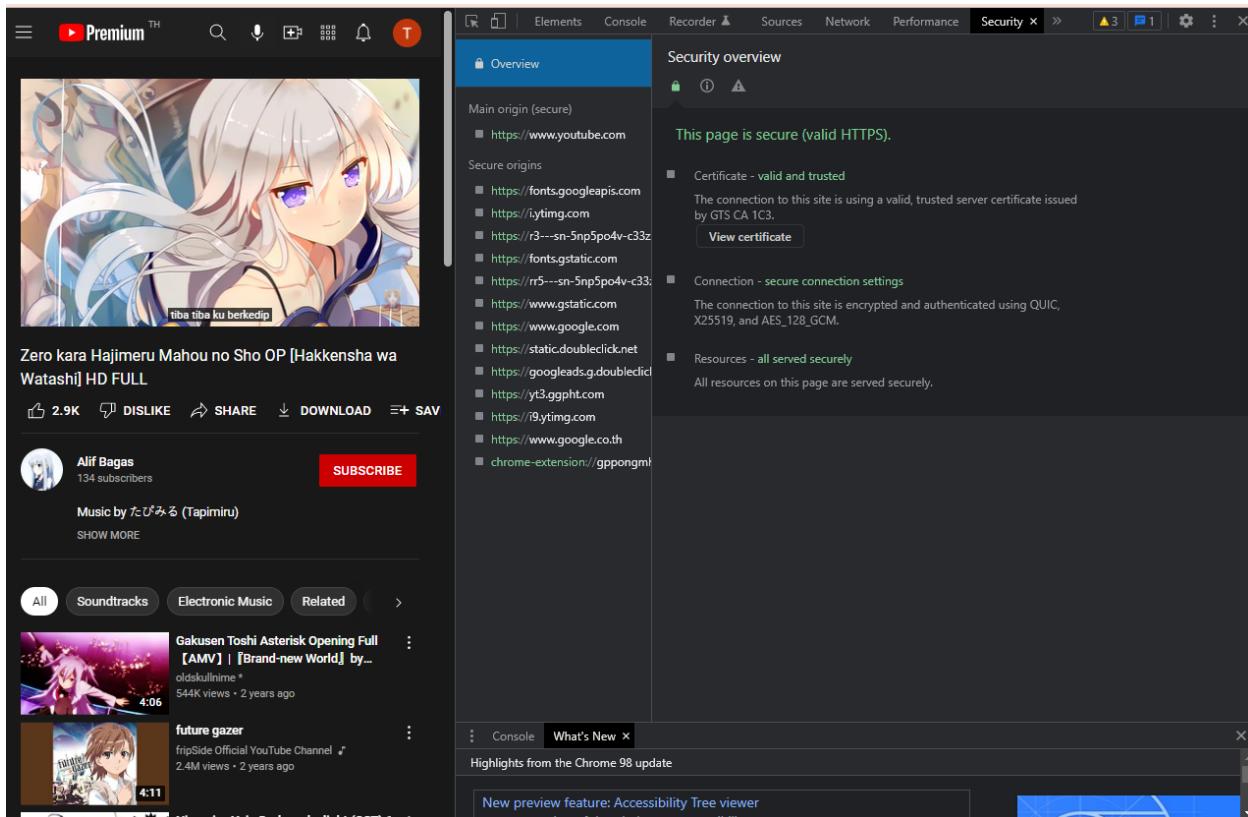
The screenshot shows the Chrome DevTools interface with the Application tab selected. On the left, there's a thumbnail of a anime girl from a video player. Below it, video thumbnails for "Zero kara Hajimeru Mahou no Sho OP [Hakkensha wa Wataishi] HD FULL" and "Sympathy - Larval Stage Planning" are visible. On the right, the Application tab displays a table of cookies for the domain <https://www.youtube.com>. The table includes columns for Name, Value, Domains, Path, Expires, Size, Height, Set-Cookie, SameSite, Path, and Priority. Some rows have checkmarks in certain columns. A message at the bottom of the table says "Select a cookie to preview its value".

Name	Value	Domains	Path	Expires	Size	Height	Set-Cookie	SameSite	Path	Priority
_Secure-3PSID...	y... /	20...	91	✓	✓	N...				H...
_Secure-3PAPI...	y... /	20...	51		✓	N...				H...
_Secure-1PAPI...	y... /	20...	51		✓					H...
SSID	y... /	20...	21	✓	✓					H...
APISID	y... /	20...	40							H...
_Secure-3PSID	y... /	20...	85	✓	✓	N...				H...
SAPISID	y... /	20...	41		✓					H...
_Secure-1PSID	y... /	20...	85	✓	✓					H...
SID	y... /	20...	74							H...
SIDCC	y... /	20...	81							H...
HSID	y... /	20...	21	✓						H...
TP_JAR	y... /	20...	19		✓	N...				M...
VISITOR_INFO1...	y... /	20...	29	✓	✓	N...				M...
_gcl_au	y... /	20...	32							M...
PREF	y... /	20...	40		✓					M...
LOGIN_INFO	y... /	20...	329	✓	✓	N...				M...
VSC	y... /	Se...	14	✓	✓	N...				M...
NID	y... /	20...	178	✓	✓					M...

รูปที่ 56 ตัวอย่างแท็บ Network

## Security

จะเป็นการสังเกตถึงความปลอดภัยของเว็บไซต์ที่ทำการสังเกตแบบคร่าว ๆ อาทิเช่น เว็บไซต์นี้ใช้ SSL หรือไม่ SSL certificate เป็นแบบใด ตอนนี้เชื่อมต่อปลอดภัยหรือไม่ เป็นต้น



รูปที่ 57 ตัวอย่างแท็บ Security ในภาพรวม

The screenshot shows a YouTube video player for "Zero kara Hajimeru Mahou no Sho OP [Hakkensha wa Watashi] HD FULL". The video thumbnail features a character with long white hair and purple eyes. Below the video, there are interaction metrics: 2.9K likes, 0 dislikes, 0 shares, 0 downloads, and 0 saves. A "SUBSCRIBE" button is visible. The page also lists related videos: "Gakuen Toshi Asterisk Opening Full [AMV] | [Brand-new World]" by oldskullname and "future gazer" by fripSide Official YouTube Channel.

On the right side, the Chrome DevTools Network tab is open, showing detailed network information for the request to "https://fonts.googleapis.com". The "Origin" section shows the main origin as "https://www.youtube.com". The "Connection" section indicates a QUIC protocol, X25519 key exchange group, and AES\_128\_GCM cipher. The "Certificate" section provides details about the SSL/TLS certificate, including the subject "upload.video.google.com", SAN "upload.video.google.com", and issuer "GTS CA 1C3". The certificate is valid from Mon, 27 Dec 2021 07:50:03 GMT to Mon, 21 Mar 2022 07:50:02 GMT.

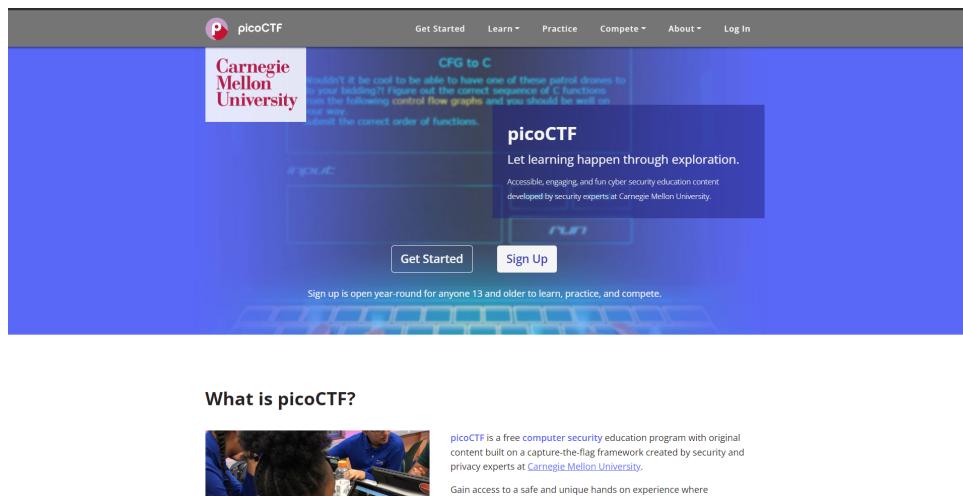
รูปที่ 58 ตัวอย่างการเลือกดูรายละเอียดความปลอดภัยในการเชื่อมต่อ

# Activity CTF

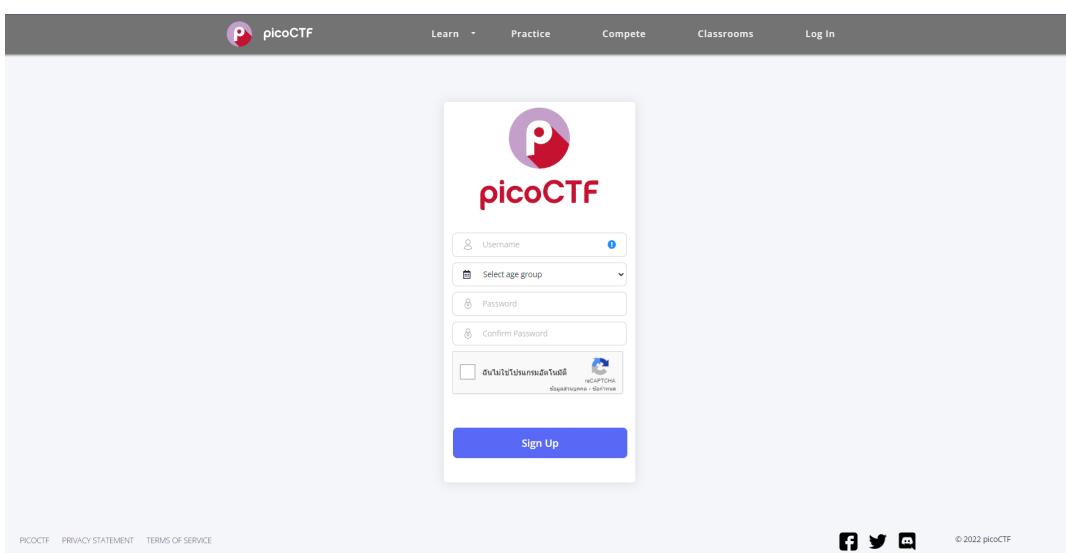
CTF ย่อมาจากคำว่า Capture The Flag เป็นเกมการแข่งขันชิงธง (Flag) โดยแต่ละทีมที่เข้าร่วมการแข่งขัน (อาจจะ 1 คนหรือมากกว่าก็ได้) จะต้องซิงธงให้ได้คะแนน มากที่สุดภายในเวลาที่กำหนด โดยคะแนนจะขึ้นอยู่ตามความท้าทายของโจทย์

## PicoCTF register

1. เข้าเว็บไซต์ <https://play.picoctf.org/>



2. สัมมารณาชิก



### 3. เลือกโจทย์ CTF กี่จะทำ

The screenshot shows the 'picoGym Practice Challenges' section of the picoCTF website. The user has a score of 350. On the left, there are filters for 'Hide Solved' and 'Show Bookmarked', and a search bar. Below that are 'Tag Filter' and 'Category Filter' sections. The 'Category Filter' is currently set to 'All Categories'. The main area displays a grid of challenges:

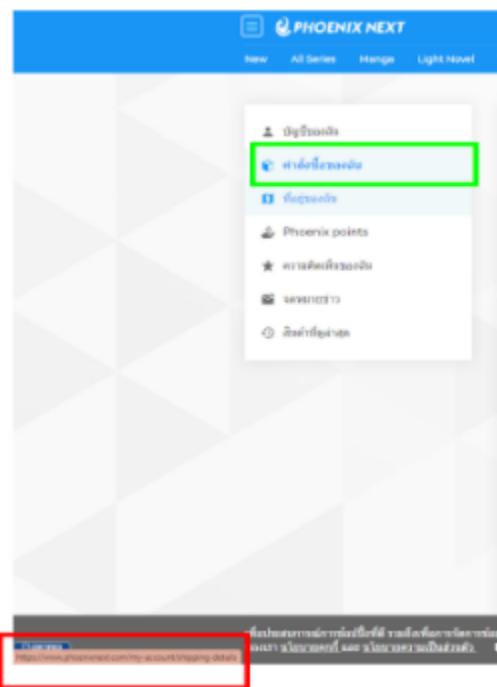
Challenge Name	Category	Points	Solves	Completion %
Obedient Cat	General Skills	5 points	68,295 solves	89% ↗
Mod 26	Cryptography	10 points	59,006 solves	91% ↗
Python Wrangling	General Skills	10 points	35,134 solves	63% ↗
Wave a flag	General Skills	10 points	42,004 solves	90% ↗
information	Forensics	10 points	26,024 solves	44% ↗
Nice netcat...	General Skills	15 points	34,147 solves	90% ↗
Transformation	Reverse Engineering	20 points	12,949 solves	56% ↗
Stonks	Binary Exploitation	20 points	6,991 solves	61% ↗
GET aHEAD	Web Exploitation	20 points	18,759 solves	81% ↗
Cryptography	Cryptography	20 points	General Skills	20 points

At the bottom of the challenge list is a separator line consisting of approximately 30 asterisks: \*\*\*\*\*

ການຜະນາກ

ໄວເປ່ອຮົງລົງກ

គីមានទីរូបនៃការស្វែងរកទិន្នន័យ និងការបង្កើតរឹងចាំនូវការ ដែលមានប៉ុណ្ណោះជាប្រព័ន្ធឌីជីថល និងការបង្កើតរឹងចាំនូវការ ដែលមានប៉ុណ្ណោះជាប្រព័ន្ធឌីជីថល



ເມື່ອນໍາເຄວົງເສຍໝາຍີ້ກີ່ດຳວ່າ “ດຳລັ້ງເຊື້ອຂອງລັນ” (ໃນກຣອບສີເຢີຍ) ຈະມີໄວ່ເປົ້ອຮັງລິງກົບປາກົງໃຫ້ເໜີນ  
ກີ່ດ້ານລ່າງ(ໃນກຣອບສີແດງ) ຜົງໄວ່ເປົ້ອຮັງລິງກົບດັ່ງກ່າວຈະພາໄປກີ່ເອກສາກີ່ໄດ້ຮູບໄວ້

สถาบันเชิร์น (CERN)

สถาบันวิจัยนิวเคลียร์แห่งยุโรป หรือเซอร์น (ตามชื่ออยู่ในภาษาฝรั่งเศสของ Conseil Européen pour la Recherche Nucléaire, CERN) เกิดจากความร่วมมือของกลุ่มประเทศในยุโรปตระวันตก 12 ประเทศ ซึ่งได้ร่วมกันจัดตั้งเพื่อเป็นองค์กรชั่วคราวในปี ค.ศ. 1952 มีหน้าที่เพื่อวิจัยและพัฒนาการก่อตั้งสถาบันวิจัยพิลิกส์นิวเคลียร์ขึ้น และการก่อตั้งได้สำเร็จลุล่วงลงใน วันที่ 29 กันยายน ค.ศ. 1954 และได้เปลี่ยนชื่อเป็น สถาบันวิจัยนิวเคลียร์แห่งยุโรป (European Organization for Nuclear Research หรือ

Organisation Européenne pour la Recherche Nucléaire ในภาษาฝรั่งเศส) แต่ยังคงใช้ชื่อเดิมขององค์กร CERN ตามเดิม สถานที่ก่อตั้งขององค์กรได้เลือกพื้นที่บริเวณตะวันตกเฉียงเหนือของเมืองเจนีวา โดยมีพื้นที่ครอบคลุมพรมแดนระหว่างasmaพันธุ์สวิตเซอร์แลนด์และประเทศฝรั่งเศส

วัตถุประสงค์ของการเชิร์นนั้นได้เขียนเอาไว้ในอนุสัญญาการก่อตั้งเชิร์น (CERN's convention) ว่าองค์กรจะต้องจัดให้มีความร่วมมือระหว่างรัฐในยุโรปในการวิจัยนิวเคลียร์ในรูปลักษณะของการศึกษา วิทยาศาสตร์ขั้นพื้นฐาน และในการวิจัยที่เกี่ยวข้องเป็นหลัก องค์กรจะต้องไม่มีความเกี่ยวข้องกับงานตามความต้องการทางทหาร และจะต้องเผยแพร่ผลการศึกษาทั้งด้านทดลองและภาคภูมิโดยการตีพิมพ์ หรือเผยแพร่โดยสาธารณะ

วัตถุประสงค์ของการเชิร์นยังคงดำเนินอยู่ เช่นเดิมจนถึงปัจจุบัน หากแต่ขอบเขตของงานวิจัยได้ปรับเปลี่ยนจากเดิมที่ต้องการศึกษาเพียงฟิสิกส์ของatom และฟิสิกส์นิวเคลียร์ เป็นงานวิจัยที่มุ่งเน้นไปยังฟิสิกส์อนุภาคมากขึ้น ซึ่งในปัจจุบันเรามีได้รู้จักเชิร์นในฐานะสถาบันวิจัยฟิสิกส์อนุภาคนี้อยู่ (European laboratory for particle physics หรือ Laboratoire européen pour la physique des particules ในภาษาฝรั่งเศส)

ในปัจจุบันเชิร์นมีจำนวนสมาชิก (Member states) รวม 23 ประเทศ และประเทศที่เป็นสมาชิกสมมทบ (Associate Members) และประเทศที่เข้าสู่การเป็นสมาชิก รวม 9 ประเทศ สมาชิกของเชิร์นในปัจจุบันไม่ได้จำกัดเฉพาะประเทศในยุโรปเท่านั้น อิสราเอลเป็นประเทศแรกนอกยุโรปที่เข้าเป็นสมาชิก ส่วนสำหรับประเทศไทยที่มีส่วนร่วมสำคัญกับโครงการสร้างพื้นฐานของเชิร์นและองค์กรที่ดำเนินงานอย่างใกล้ชิดกับเชิร์นเป็นผู้สัมภาระ重任 ได้แก่ ประเทศไทย สหพันธ์รัฐรัสเซีย สหรัฐอเมริกา สหภาพยุโรป (EU) สถาบันร่วมวิจัยนิวเคลียร์ (JINR) และองค์การเพื่อการศึกษา วิทยาศาสตร์ และวัฒนธรรมแห่งสหประชาชาติ (UNESCO)

## ไคลเอนต์ - เชิร์ฟเวอร์ (Client - Server)

ไคลเอนต์ - เชิร์ฟเวอร์ (Client - Server) คือ การที่มีเครื่องผู้ใช้บริการ (Server) และเครื่องผู้ใช้บริการ (Client) เข้ามาร่วมกันอยู่ และเครื่องผู้ใช้บริการได้มีการติดต่อร้องขอข้อมูลจากเครื่องผู้ใช้บริการ เครื่องผู้ใช้บริการจะจัดการตามที่เครื่องผู้ใช้บริการร้องขอ และส่งข้อมูลกลับไปให้เครื่องผู้ใช้บริการ

## just-in-time compilation

ในภาษาโปรแกรม Java และสภาพแวดล้อม just-in-time (JIT) compiler คือโปรแกรมที่เปลี่ยนกลับ bytecode ของ Java ให้เป็นคำสั่งที่สามารถส่งตรงไปที่โปรแกรมเซอร์ หลังจากการเขียนโปรแกรม ภาษา Java คำสั่งต้นแบบจะได้รับคอมไพล์ โดย Java compiler เป็น bytecode แทนที่จะ

เป็นรหัสที่เก็บคำสั่งที่ต้องกันแพล็ตฟอร์มของโปรแกรม เช่น Intel Pentium, IBM System /390 bytecode เป็นรหัสที่ไม่ขึ้นกับแพล็ตฟอร์มที่สามารถส่งไปยังแพล็ตฟอร์มต่าง ๆ และทำงานบนแพล็ตฟอร์มนั้น ซึ่งในอดีตโปรแกรมส่วนมากที่เขียนในภาษาต่าง ๆ จะต้องมีการคอมไพล์ช้า และบางครั้งต้องเขียนใหม่สำหรับแต่ละแพล็ตฟอร์ม ทำให้ข้อได้เปรียบอย่างมากของ Java คือสามารถเขียนและคอมไพล์เพียงครั้งเดียว Java ในแต่ละแพล็ตฟอร์มจะแปลง bytecode เป็นคำสั่งที่เข้าใจได้ของโปรแกรม เช่น

## Rules of Request Methods

- ไม่ใช้ Method GET และ POST เพื่อกดแทนการทำ Action แบบอื่น ๆ ที่สามารถทำผ่าน HTTP Method — ได้ เช่น Method เป็น POST แต่และใน Action DELETE เข้าไปใน Header หรือ Body เพื่อให้ Server ทำการลบข้อมูล เพราะการทำแบบนี้จะทำให้เกิดความเข้าใจผิด และสูญเสียความโปร่งใสของ HTTP Protocol

- Method GET จะถูกใช้เพื่อรับข้อมูลหรือ ตัวแทนของ Resource ที่ Client ต้องการ เก่านั้น — ส่วนใหญ่แล้วการทำงานของ Web จะหนักไปกว่า GET Method ดังนั้น Client ควรจะวางแผนให้ได้ เมื่อเรียก GET ช้า ๆ ต้องทำได้โดยไม่มีผลกระทบอะไร ซึ่งมีผลให้เราใช้ความสามารถของ Cache ได้ ซึ่งจะเป็นการที่จะส่งข้อมูลเดิมกลับไปที่ Client โดยที่ไม่ต้องต่อ กับ Server จริง ๆ (ซึ่งขึ้นกับความต้องการว่าต้องการข้อมูลที่กันสมัยแค่ไหน ข้อมูลมีการเปลี่ยนแปลงบ่อยไหม และข้อมูลต้องถูกต้องตามข้อมูลที่เก็บจริง ๆ หรือเปล่า เพราะถ้าต้อง Query Data จาก Database ตลอดเวลา ก็อาจจะไม่ใช่เรื่องดีนักสำหรับการรองรับ Scale ใหญ่)

- Method HEAD ถูกใช้เพื่อดูข้อมูล Response Header — Client จะใช้ Head ก็ต้องเมื่อต้องการข้อมูลที่อยู่ใน Response Header โดยไม่สนใจ Body เช่น การ เช็ค Last-Modified เพื่อต้องว่าต้องดึงข้อมูลใหม่หรือยัง หรือจะใช้ดู Content-Length ที่อยู่ใน Header ก็ได้ เพราะฉะนั้น Method HEAD ก็เหมือนกับ GET เพียงแต่ไม่เอา Body เก่านั้น

- Method PUT จะถูกใช้เพื่อการ Insert และ Update เข้าไปใน Resource ประเภท Store — PUT ถูกใช้ในการสร้าง Resource ใหม่ใน Store ด้วยการ Client จะต้องระบุ URI ของ Resource (เพียงตัวเดียว และระบุ Identifier) ที่ชัดเจน เช่น /users/12345 รวมถึงถูกใช้ในการเปลี่ยนแปลงแก้ไข Resource นั้น ๆ ด้วย เช่นจาก Resource ที่ถูกสร้างจากการ PUT ถูกส่งมาจาก Client นั้นไม่ได้แปลว่า ข้อมูลที่ PUT ขึ้นมากับ ข้อมูลที่ถูก GET หลังจากนั้น (ด้วย ID เดียวกัน) จะต้องเหมือนกันเสมอ ยกตัวอย่างเช่น การที่เราจะจะยอมให้ Client ปรับเปลี่ยนข้อมูลที่เข้าสามารถทำได้ และส่งมาใน Request เท่านั้น จุดสำคัญอีกอย่างของ PUT คือการตกลงกันของกรณีเกิด Conflict เนื่องจากเราใช้ PUT เพื่อ Insert และ Update การที่ Client 2 คนจะส่ง Request กับ Id เดียวกันมากที่ REST นั้นอาจจะหมายถึงการกันกันไปเรื่อย ๆ หรืออาจจะหมายถึงการที่พยายามสร้างข้อมูลซ้ำกันก็ได้ (อาจจะเพิ่ม Header เพื่อบอกเช่น If-Unmodified-Since, If-Match เป็นต้น)

- Method PUT จะถูกใช้เพื่อเปลี่ยนแปลงข้อมูลเดิมของ Resource ได้ — นั่นแปลว่า Client ต้องบอกมาว่าจะเปลี่ยนแปลงอะไร

- Method POST จะถูกใช้เพื่อการสร้าง Resonurce ใหม่ใน Collection ที่กำหนด — Client จะใช้ POST ในการสร้าง Resource ใหม่ไปยัง Collection ที่ต้องการ เช่น /users ซึ่ง Request Body ของ POST จะแสดงข้อมูลที่ใช้เป็นตัวแทนของ Resource ไปยัง Server ที่จะเก็บ Resource นั้น ๆ ได้ เช่นอาจจะส่ง UUID ไปให้ Server เพื่อใช้เป็น Id ของ Resource นี้ต่อไป ซึ่งโดยปกติในกรณีของ POST หน้าที่ Initial ค่าทางค่าของ Resource มักจะทำที่ Server ให้มองว่าเราทำ POST ไปที่ Webboard เราไม่สร้าง Id เองก่อน POST แน่นอน

- Method POST จะถูกใช้เพื่อเพื่อประมวลผลคำสั่งอื่น ๆ ที่ไม่ใช่

CRUD — Client จะใช้ POST สำหรับเรียกใช้ Function กีดูบคุณการทำงานของ Resource กีนอกเหนือจาก CRUD Function ซึ่งนั่นแปลว่าจะต้องส่งข้อมูลตาม Function นั้นมาในรูปแบบของ Header หรือใน Body ด้วย Protocol HTTP ได้ออกแบบให้ POST เป็นลักษณะของ Method ปลายเปิด เพราะวัน POST จึงถูกใช้เป็น Operation อะไรก็ได้ ที่ไม่ไปข้ามกับ การขออุด การเก็บ และการลบ Resource และด้วย HTTP Protocol ระบุให้ POST Request เป็น unsafe และ non-idempotent (อาจจะไม่ใช้ผลลัพธ์เป็นค่าเดิมแม้ว่าจะกระทำการดำเนินการดังกล่าวซ้ำอีกรอบ) นั่นแปลว่าเราคาดผลลัพธ์ที่จะเกิดขึ้นไม่ได้ และไม่รับประกันว่าการทำซ้ำ สามารถทำได้โดยไม่มีผลข้างเคียง เช่น การส่งคำสั่งซื้อซ้ำ ๆ ผ่าน POST ก็จะมีผลใช้การตัดบัตรเครดิตของลูกค้าเกิดขึ้นมากกว่า 1 ครั้งได้ (ซึ่งอันนี้ต้องจัดการเอง)

- Method DELETE จะถูกใช้เพื่อลบ Resource ออกจาก

Collection หรือ Store กีเก็บมัน — DELETE Method ต้องระบุ Resource กีต้องการลบ เช่น DELETE /users/1234 และว่าต้องการลบ Resource 1234 ใน Collection กีชื่อ Users ซึ่งหลังจากนั้นเมื่อกำการเรียกใช้ GET หรือ HEAD มาที่ Resource ดังกล่าวจะต้องได้ Status Code 404 ("Not Found") กลับไปใน HTTP Protocol

- Method OPTIONS อาจจะถูกใช้เพื่ออ่าน Metadata ของ

Resource ว่ารองรับการทำงานอะไรบ้าง — เช่น Client อาจจะใช้ OPTIONS ไปยัง Server และได้รับการตอบกลับมาใน Response Header ว่า Allow: GET, PUT, DELETE เป็นต้น ซึ่งในบางกรณีเราอาจจะส่ง Link ของแต่ละ Operation กลับไปใน Body ด้วยก็ได้

ข้อแตกต่างระหว่าง PUT Method และ POST Method คือ PUT Method ทำโดยตรงกับ Resource โดยต้องระบุ Id เข้าไปจาก Client ซึ่งนั่นแปลว่า การจัดการสร้าง Resource ของ PUT Method ทำที่ Client ส่วน POST จะบุ๊ฟ Collection และในฝั่ง Server จะสร้าง Id ขึ้นมาเพื่อ Response ให้ Client เพื่อเอาไปใช้อีกที่ นั่นแปลว่าการสร้าง Resource ทำโดย Server

จุดอื่น ๆ ที่สำคัญคือ PUT Method เป็น Idempotent ซึ่งแปลว่าทำซ้ำกีครั้งก็ได้ผลเหมือนเดิม และสามารถ Cache Response กีเกิดขึ้นได้ ส่วน POST Method สร้าง Id ใหม่ทุกครั้งนั่นแปลว่ารับแต่ละครั้งจะได้ผลไม่เหมือนเดิม

## Request for Comments (RFC)

เอกสารคำขอความเห็นถูกใช้โดยชุมชนอินเทอร์เน็ตมานานกว่า 40 ปีเพื่อกำหนดมาตรฐานใหม่ ๆ และแบ่งปันข้อมูลทางเทคโนโลยี นักวิจัยจากมหาวิทยาลัยและ บริษัท ต่าง ๆ เผยแพร่เอกสารเหล่านี้เพื่อเสนอแนวทางปฏิบัติที่ดีที่สุดและขอความคิดเห็นเกี่ยวกับเทคโนโลยีอินเทอร์เน็ต RFCs ได้รับการจัดการโดยองค์กรที่เรียกว่า Internet Engineering Task Force

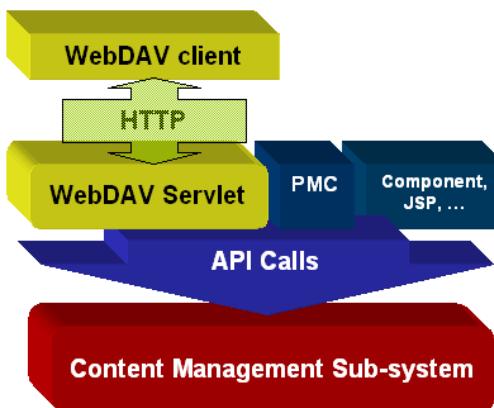
RFCs แรกที่รวมถึง RFC 1 ถูกตีพิมพ์ในปี 2512 และว่าเทคโนโลยีในยุคหนึ่น "ซอฟต์แวร์โอเพนซ์" กีกล่าวถึงใน RFC 1 นับแต่นั้นมาล้าสมัยแล้ว เอกสารเช่นนี้ก็ได้ให้ความสำคัญกับการเริ่มต้นของระบบเครือข่ายคอมพิวเตอร์ แม้แต่วันนี้รูปแบบข้อความธรรมดาก่อตัวของ RFC ยังคงเป็นหลักเหมือนเดิมตั้งแต่ต้น

เทคโนโลยีเครือข่ายคอมพิวเตอร์ที่ได้รับความนิยมในช่วงแรกของการพัฒนาได้รับการจัดทำเป็นเอกสารไว้ใน RFCs ตลอดหลายปีที่ผ่านมา เช่น แนวคิดชื่อโอดเมนอินเทอร์เน็ต (RFC 1034), การจัดสรรที่อยู่สำหรับ อินเทอร์เน็ต ส่วนตัว (RFC 1918), HTTP (RFC 1945), HTTP (RFC 1945) IPv6 (RFC 2460) เป็นต้น

แม้ว่าเกตโนโลยีขั้นพื้นฐานของอินเทอร์เน็ตจะครบถ้วนแล้ว กระบวนการ RFC ยังคงทำงานผ่าน IETF อยู่ เอกสารมีการร่างและดำเนินการผ่านขั้นตอนต่าง ๆ ของการทบทวนก่อนที่จะให้สัตยาบันขั้นสูงท้าย หัวข้อที่ครอบคลุมใน RFCs มีไว้สำหรับผู้มีงานวิจัยระดับมืออาชีพและวิชาการที่มีความเชี่ยวชาญสูง แทนที่จะโพสต์ความคิดเห็นสาธารณะแบบ Facebook ความคิดเห็นเกี่ยวกับเอกสาร RFC จะได้รับผ่านทางเว็บไซต์ RFC Editor แทน มาตรฐานขั้นสูงท้ายจะเผยแพร่ในเดือนนี้หลัง RFC ที่ rfc-editor.org

## WebDAV

WebDAV ย่อมาจาก Web-based Distributed Authoring and Versioning เป็นโปรโตคอลที่ทำให้สามารถจัดการ อ่านและเขียนไฟล์บนเว็บเซิร์ฟเวอร์ได้ เพื่อให้ผู้ใช้งานสามารถมีส่วนร่วมในการจัดการทรัพยากร่างต่าง ๆ บนเครื่อง Server ใน การจัดการทรัพยากรเหล่านั้น ผู้ใช้งานจะทำอยู่บน Protocol HTTP โปรแกรมที่ใช้งานกันก็คือ Outlook โปรแกรมอ่าน E-Mail ของ Microsoft โดยการดึงเมล์มาจาก hotmail



## XML (Extensible Markup Language)

เป็นภาษาหนึ่งที่ใช้ในการแสดงผลข้อมูล ซึ่งภาษาที่ใช้กำหนดรูปแบบของคำสั่งภาษา HTML หรือที่เรียกว่า Meta Data ซึ่งจะใช้สำหรับกำหนดรูปแบบของคำสั่ง Markup ต่าง ๆ ถ้าเปรียบเทียบกับภาษา HTML จะแตกต่างกันที่ HTML ถูกออกแบบมาเพื่อการแสดงผลอย่างเดียวเท่านั้น เช่นให้แสดงผลตัวเล็ก ตัวหนา ตัวเอียง แต่ภาษา XML นั้นถูกออกแบบมาเพื่อเก็บทั้งข้อมูลและโครงสร้างของข้อมูลนั้น ๆ ໄ้ด้วยกัน ส่วนการแสดงผลก็จะใช้ภาษาเฉพาะซึ่งก็คือ XSL (Extensible Stylesheet Language)

ภาษา XML มีโครงสร้างที่ประกอบด้วยแท็กเปิด และแท็กปิด เหมือนเดียวกับภาษา HTML แต่ภาษา XML สามารถสร้างแท็กรวมทั้งกำหนดโครงสร้างของข้อมูลได้เอง ซึ่งความสามารถตรงนี้ตัวภาษา HTML ทำไม่ได้ เพราะภาษา HTML ถูกกำหนดโดย W3C หรือ World Wide Web Consortium

อาจกล่าวได้ว่า XML เป็นส่วนเสริมของ HTML เพราะตัว XML ไม่สามารถแสดงผลได้ในตัวของมันเอง หากต้องการแสดงผลที่ถูกต้อง จะต้องมีการใช้ร่วมกับภาษาอื่น เช่น HTML, JSP, PHP, ASP หรือภาษาอื่น ๆ ที่สนับสนุน XML จะมีนามสกุลเป็น .XML สามารถสร้างขึ้นจากโปรแกรมประเภท Text Editor ได้ ก็ได้ เช่น Notepad, Editplus , DreamWeaver, MS Word เป็นต้น

## Nginx

Nginx (Engine-X) คือ Web Server ที่นิยมใช้ในปัจจุบัน โดยผู้ใช้งานกล่าวกันว่าสามารถรองรับการใช้งานหลากหลายกว่า Apache และมีประสิทธิภาพการทำงานสูง อีกทั้งยังมีโมดูลเสริมให้ใช้งานอย่างเพียงพอต่อการใช้งาน ที่ดีไปว่านั้นคือเป็นซอฟต์แวร์แบบ Open source ทำให้สามารถใช้งานได้แบบไม่มีค่าใช้จ่ายนั้นเอง ตัวระบบมีให้เลือกรองรับทั้งแบบ Linux และ Windows

Nginx ถูกพัฒนาขึ้นมาเมื่อปี 2547 โดย อิ戈อร์ ซิลโซเยฟ (Igor Sysoev) มีการทดลองแก้ไขหลาย ๆ อย่างก่อนจะปล่อยออกมายังใช้งาน จึงมีความเสถียรสูง กินทรัพยากระบบทน้อย เริ่มต้นถูกใช้งานอย่างมากในรัสเซีย ก่อนที่จะมีการกำคู่มือภาษาอังกฤษและเริ่มเผยแพร่โดยใช้งานกันไปทั่วโลก ส่วนใหญ่แล้วผู้ใช้งาน Nginx จะถูกใช้งานกับพาร์ทิชันที่มีการอัพโหลด ดาวโหลดบ่อย ๆ หรือใช้ในการสตรีมมิ่ง เนื่องด้วย Nginx กินทรัพยากร้น้อยทั้ง ram และ cpu ทำให้สามารถเรียกการทำงานได้มากยิ่งขึ้น

### หลักการทำงานของ Nginx

การใช้งานของ Nginx จะใช้งาน Nginx Web Server ควบคู่กับ Apache Web Server โดยให้ Nginx รับ request และส่งไปยัง Apache เพื่อประมวลผลการทำงาน PHP เพื่อแสดงผล

## Status Code และ Reason Phrase เพิ่มเติม

### Informational 1xx (ข้อมูล)

จะใช้เพื่อบอก Response ข้อควรรู้ ซึ่งมีส่วนประกอบเพียง Status-Line และ Headers เท่านั้น โดยปกติแล้วนั้นจะไม่มีการใช้รหัส 1xx ใน การใช้งานทั่วไปโดยตรง เพราะว่าในตัวของ HTTP/1.0 จะไม่ได้กำหนดรหัส 1xx เอาไว้ (จะอยู่ใน HTTP/1.1) แต่จะใช้ในการบอกว่า เซิร์ฟเวอร์ที่กำลังดำเนินการของไคลเอนต์ไปประมวลผลอยู่เป็นอย่างไรบ้าง โดยจะมีรายละเอียดดังนี้

### 100 Continue

Request ที่ได้รับการตอบกลับมาสามารถใช้งานได้ และควรจะใช้ Request นี้ต่อไปหากว่ายังต้องการติดต่ออยู่

### 101 Switching Protocols

User Agent ทำการร้องขอให้เซิร์ฟเวอร์ทำการเปลี่ยนโพรโตคอลในการติดต่อ

## 102 Processing (WebDAV; RFC[Request For Comment] 2518)

รหัสนี้ระบุว่าเซิร์ฟเวอร์ได้รับคำขอ **WebDAV** และกำลังดำเนินการ คำขอ WebDAV สามารถมีคำขออยู่อย่างจำนวนมากอยู่ภายนอก และอาจใช้เวลานานในการดำเนินการตามแอปพลิเคชัน รหัสนี้สามารถป้องกันไม่ให้คลื่นทึบเวลาโดยสันนิษฐานว่าคำขอจะหายไป

### Successful 2xx (สำเร็จ)

หมายความว่าการ Request นั้นได้รับแล้วและกระทำการตาม Method สำเร็จโดย Server โดยจะมีรายละเอียดดังนี้

#### 200 OK

เป็นมาตรฐานของ HTTP Response นั้น Success สำหรับ GET, PUT หรือ POST

#### 201 Create

เป็น Response สำหรับข้อมูลใหม่ที่ถูกสร้างขึ้น ใช้สำหรับ POST

#### 202 Accepted

Request นั้นได้ถูกยอมรับให้นำไปประมวลผลได้ แต่การประมวลผลยังไม่เสร็จสิ้น

#### 203 Non-Authoritative Information

เป็นการบอกว่าข้อมูลที่ส่งกลับมาต้นไม่ใช่ข้อมูลที่มาจากเซิร์ฟเวอร์ต้นทาง แต่รวมมาจากข้อมูลที่มีในเครื่องหรือมาจากสำเนาของบุคคลที่สาม (Third-Party Copy)

#### 204 No Content

เป็น Response สำหรับ Request ที่ดำเนินการ Success แต่ไม่ได้ return ข้อมูลกลับ

#### 205 Reset Content

เซิร์ฟเวอร์ได้ดำเนินการตาม Request และ User Agent ควรรีเซ็ตมุมมองเอกสาร ซึ่งทำให้มีการส่ง Request นั้นๆ โดย Response นี้มีจุดประสงค์หลักเพื่อจัดการข้อมูลนำเข้าเพื่อใช้ในการประมวลผลที่เกิดขึ้นผ่านการป้อนข้อมูลของผู้ใช้ ตามด้วยการล้างแบบฟอร์มที่ให้ข้อมูลนำเข้าเพื่อให้ผู้ใช้สามารถเริ่มต้นการดำเนินการนำข้อมูลอื่นเข้าได้จ่ายอีก

#### 206 Partial Content (RFC 7233)

เป็นการระบุว่าเซิร์ฟเวอร์ได้ดำเนินการตาม Request ไปบางส่วนแล้ว และ Request มีข้อมูลบางอย่างที่ต้องการการ Response ตามเงื่อนไขบางอย่างที่ต้องของ Request ระบุไว้

### 207 Multi-Status (WebDAV; RFC 4918)

เป็นการระบุว่าการตอบสนองที่ตามมาเป็น XML Message ซึ่งสามารถรวม Response Code ที่ไม่ต่อเนื่องได้จำนวนหนึ่ง การตอบสนองเหล่านี้ขึ้นอยู่กับจำนวนคำขออย่างที่เริ่มต้นโดยได้ล่อนต์

### 208 Already Reported (WebDAV; RFC 5842)

เป็นการระบุว่า DAV นั้นเคยนับรวมและแสดงรายการในการตอบกลับคำขออนหนานี้ และไม่ได้รวมเข้าด้วยกันอีกต่อไป

### 226 IM Used (RFC 3229)

หมายความว่า เซิร์ฟเวอร์ได้ดำเนินการตามคำขอสำหรับทรัพยากรแล้วอย่างน้อยหนึ่งรายการที่ได้ทำการร้องขอไป

### Redirection 3xx (เปลี่ยนเส้นทางใหม่)

เป็นการบอกว่า User Agent จำเป็นต้องดำเนินการเพิ่มเติมเพื่อดำเนินการตาม Request การดำเนินการที่ว่าอาจจะดำเนินการโดย User Agent เองโดยไม่ต้องโต้ตอบกับผู้ใช้ชี้อีกครั้ง ซึ่งมักจะเป็นการเปลี่ยนเส้นทางการเข้าถึงใหม่ (อาจจะเปลี่ยนไปที่หน้าเว็บเพจอื่นหรือเว็บไซต์อื่นก็ได้) โดยจะมีรายละเอียดดังนี้

### 300 Multiple Choices

ในบางครั้ง เซิร์ฟเวอร์อาจมีแหล่งข้อมูลที่เป็นไปได้หลายอย่างเพื่อตอบสนองคำขอของ User Agent รหัสนี้จะหมายความว่า User Agent ต้องเลือกระหว่างตอนนี้ ซึ่งอาจเกิดขึ้นเมื่อมีนามสกุลไฟล์หลายนามสกุล หรือหากเซิร์ฟเวอร์กำลังประสบกับความของคำที่ดู kaum

### 301 Moved Permanently

ทรัพยากรที่ Request ได้ทำการร้องขอมาได้ถูกเปลี่ยน URI (Uniform Resource Identifier) ใหม่อย่างถาวร โดยอาจจะถูกเปลี่ยนเส้นทาง (URL[Uniform Resource Locator] Redirect) หลังจากนี้หรือไม่ก็ได้

### 302 Found

ทรัพยากรที่ Request ได้ทำการร้องขอมาถูกทำการเคลื่อนย้าย แต่ยังสามารถพบได้อยู่ (มักจะพบในตำแหน่งที่ไม่ได้คาดไว้) โดยจะเป็นสิ่งที่เกิดในการเปลี่ยนเส้นทางชั่วคราว (Temporary URL Redirection)

### 303 See Other (HTTP/1.1 ขึ้นไป)

เป็นการบอกว่าพบรัพยากรที่ Request ได้ทำการร้องขอมา แต่่ว่า การทำการ Request Method ก็อยู่ใน Request Message ที่ได้ร้องขอมาควรใช้ URL อื่นก็ไม่ใช่ URL นี้

### 304 Not Modified (RFC 7232)

เป็น status code ก่อนกว่า client ได้รับการ Response และอยู่ใน cache และไม่จำเป็น จะต้องส่งผ่านข้อมูลเดิมอีกครั้ง

### 305 Use Proxy (HTTP/1.1 ขึ้นไป)

กรรพยากรที่ Request ได้ทำการร้องขอมาหนึ่นต้องเข้าถึงผ่าน Proxy (ตัวกลาง) ที่จะระบุ มาให้เท่านั้น

### 306 Switch Proxy(Unused)

เป็นการระบุว่า Request ที่ได้ร้องขอมาควรกลับไปที่ Proxy ที่ระบุไว้ ซึ่งเคยถูกใช้ใน HTTP เวอร์ชันก่อนหน้าของแต่ในปัจจุบันไม่ใช้แล้ว และรหัสลับนี้ยังถูกสงวนเอาไว้อีกด้วย

### 307 Temporary Redirect (HTTP/1.1 ขึ้นไป)

พบกรรพยากรที่ได้ร้องขอใน URI อื่นได้ แต่ใน URI เดิมก็ยังพบได้ เช่นกัน Status Code นี้อาจจะคล้ายคลึงกับ Status Code 302 เพียงแต่ว่า Status Code 307 สามารถเปลี่ยนแปลง ตัว HTTP Method ได้หากกรรพยากรที่ได้ร้องขอได้ถูกเคลื่อนย้าย

### 308 Permanent Redirect (RFC 7538)

กรรพยากรที่ได้ร้องขอหนึ่นควรไปร้องขอใน URI อื่น Status Code นี้อาจจะคล้ายคลึงกับ Status Code 301 เพียงแต่ว่า Status Code 308 สามารถเปลี่ยนแปลงตัว HTTP Method ได้

### Client error 4xx (ความผิดพลาดในฝั่งของไคลเอนต์)

โดย Status ในกลุ่มนี้จะระบุถึงความผิดพลาดในฝั่งของไคลเอนต์ที่ทำให้ Request ที่ได้ทำการร้องขอมาไม่สามารถดำเนินการให้สำเร็จได้ โดยจะมีรายละเอียดดังนี้

### 400 Bad Request

Request กี่ส่งมาโดย Client นั้นไม่ถูกดำเนินการ และ Server ไม่เข้าใจว่า Request ตั้งกล่าต้องการร้องขอเกี่ยวกับอะไร

#### 401 Unauthorized (RFC 7235)

Client ไม่ได้รับอนุญาตในการเข้าถึงข้อมูลและควรจะส่ง Credential(หนังสือรับรองหรือสิ่งยืนยันลิขิตร์) มาพร้อม request

#### 402 Payment Required

ยังไม่ได้ถูกใช้ในปัจจุบัน แต่ได้ถูกสงวนเอาไว้สำหรับการใช้งานในอนาคต

#### 403 Forbidden

บ่งบอกว่า Request นั้นถูกต้องและ Client ได้รับการอนุญาต แต่ Client ไม่ได้รับการอนุญาตให้เข้าถึงข้อมูลด้วยเหตุผลบางประการ

#### 404 Not Found

บ่งบอกว่า Resource กี่ Request มาแล้ว ยังไม่สามารถใช้งานได้ในขณะนี้ แต่ไฟล์อินเทอร์เน็ตที่เข้าถึงข้อมูลนี้ได้

#### 405 Method Not Allowed

HTTP Method กี่ระบุอยู่บน Request-Line ไม่ถูกอนุญาตให้ใช้บน Request-URI ของกรัพยากรที่ถูกต้องขอ

#### 406 Not Acceptable

กรัพยากรที่ถูกต้องขอมาแล้วไม่สามารถสร้างเนื้อหาได้ตามที่ Request สามารถรับได้ (ไม่ตรงตามที่ระบุไว้บน Accept Header ใน Request Message)

#### 407 Proxy Authentication Required (RFC 7235)

คล้ายคลึงกับ Status Code 401 (Unauthorized) เพียงแต่ว่าไฟล์อินเทอร์เน็ตต้องระบุตัวตนกับ Proxy กี่ตนเองใช้ก่อนเป็นอันดับแรก

#### 408 Request Timeout

ไฟล์อินเทอร์เน็ตไม่สามารถส่งคำขอ (Request) ภายในระยะเวลาที่เซิร์ฟเวอร์เตรียมพร้อมกี่จะรอรับคำขอได้กัน ซึ่งไฟล์อินเทอร์เน็ตสามารถส่งคำขอไปใหม่ได้ในภายหลัง

## 409 Conflict

คำขอ (Request) ไม่สามารถทำให้สำเร็จได้เนื่องจากเกิดข้อขัดแย้งกับสถานะปัจจุบันของทรัพยากรที่ร้องขอ

## 410 Gone

ทรัพยากรที่ร้องขอนั้นไม่มีอยู่แล้ว ซึ่งอาจจะคล้ายคลึงกับ Status Code 404 เพียงแต่ว่าใน Status Code 410 นี้จะหมายถึง ทรัพยากรที่ร้องขออาจถูกย้ายหรือถูกลบไปอย่างถาวร

## 411 Length Required

เซิร์ฟเวอร์ไม่ตอบรับคำขอที่ไม่ได้ระบุความยาวของเนื้อหา (Content- Length)

## 412 Precondition Failed (RFC 7232)

User Agent ได้มีเงื่อนไขในส่วนหนึ่งของคำขอ (Request Header) ซึ่งเซิร์ฟเวอร์ปลายทางไม่ตรงตามเงื่อนไขเหล่านั้น

## 413 Request Entity Too Large [Payload Too Large] (RFC 7231)

เซิร์ฟเวอร์ไม่ตอบรับคำขอที่มีเนื้อหาที่ยาวเกินกว่าเซิร์ฟเวอร์จะดำเนินการต่อได้ ซึ่งเซิร์ฟเวอร์อาจขยายด้วยการเชื่อมต่อกับโคลอีน์ต์ก่อนกีโคลอีน์ต์จะส่งคำขอต่อ

## 414 Request-URI Too Long (RFC 7231)

เซิร์ฟเวอร์ไม่ตอบรับคำขอสำหรับการร้องขอทรัพยากรที่มี URI ยาวเกินกว่าเซิร์ฟเวอร์จะดำเนินการต่อได้

## 415 Unsupported Media Type

เซิร์ฟเวอร์ไม่ตอบรับคำขอสำหรับการร้องขอเนื้อหาประเภทที่เซิร์ฟเวอร์ไม่สนับสนุน

## 416 Requested Range Not Satisfiable

ทรัพยากรที่ได้ทำการร้องขอเป็นส่วนหนึ่งของทรัพยากรที่ไม่สามารถเรียกได้

## 417 Expectation Failed

เซิร์ฟเวอร์ไม่พบ Expect request-header (ถูกระบุไว้ที่ Request Header)

## 418 I'm a teapot (RFC 2324)

Status Code จะดีนค่ากลับมาเป็น “ถัวยน้ำชา” ซึ่งเป็นมุกตลกของ วัน April Fool's Joke ในปี 1998.



418. I'm a teapot.

The requested entity body is short and stout.  
Tip me over and pour me out.



## 422 Unprocessable Entity (WebDAV; RFC 4918)

คำขอของไคลเอนต์มีความผิดพลาดและเซิร์ฟเวอร์ไม่สามารถประมวลผลได้

## 423 Locked (WebDAV; RFC 4918)

ข้อมูลที่ถูกเรียกขอถูกล็อคเอาไว้

## 424 Failed Dependency (WebDAV; RFC 4918)

เกิดขึ้นการร้องขอที่ทำขึ้นล้มเหลวเนื่องจากความล้มเหลวของการร้องขอ ก่อนหน้าการร้องขอันนั้น ๆ

## 425 Too Early

เกิดขึ้นเมื่อเซิร์ฟเวอร์ไม่ประมวลผลคำขอของผู้ใช้ไคลเอนต์ เนื่องจากอาจจะเป็นการส่งคำขอเดิมมาช้า อีกครั้งหนึ่ง

## 426 Upgrade Required

เนื่องจากเนื้อหาของ Request Header ได้ทำการอัปเกรด โคลเลนต์ควรเปลี่ยนไปใช้ไปร์โടคอลอื่น

#### 428 Precondition Required (RFC 6585)

เซิร์ฟเวอร์ได้ทำการกำหนดให้ต้องระบุเงื่อนไขก่อนดำเนินการตามคำขอ

#### 429 Too many requests (RFC 6585)

เกิดขึ้นเมื่อผู้ใช้ส่งคำขอมากเกินไปในระยะเวลาที่กำหนด โดยเซิร์ฟเวอร์อาจจะใช้เพื่อป้องกันบอทหรือสคริปต์ที่พยายามจะเข้าถึงเว็บไซต์ของตนเอง

#### 431 Request Header Fields Too Large (RFC 6585)

เซิร์ฟเวอร์ไม่สามารถประมวลผลคำขอได้เนื่องจากล้วนหัวของคำขอ (Header Field) มีขนาดใหญ่เกินไป ซึ่งในบางครั้งอาจจะหมายถึงแค่ล้วนหัวหรือทั้งหมดเลยก็ได้ เช่นกัน

#### 451 Unavailable for Legal Reasons

ผู้ให้บริการเซิร์ฟเวอร์ได้รับคำขอให้ห้ามการเข้าถึงกรัพยากรหรือชุดของกรัพยากรที่ได้ทำการร้องขอ

#### 499: "Client closed request."

ถูกใช้โดย Nginx เมื่อโคลเลนต์ได้ทำการปิดคำขอ ในตอนที่ Nginx ยังทำงานอยู่ Server Error 5xx (ความผิดพลาดในฝั่งของเซิร์ฟเวอร์)

ในล้วนของ 500 Internal Server Error บอกว่าการ request นั้นถูกต้อง แต่ Server มีความสับสนและจะบริการด้วยเงื่อนไขดังกล่าวไม่ได้ ซึ่งจะมีรายละเอียดดังนี้

#### 501 Not Implemented

เซิร์ฟเวอร์ไม่รองรับฟังก์ชันที่จำเป็นในการดำเนินการตามคำขอ ซึ่งจะเกิดขึ้นเมื่อเซิร์ฟเวอร์ไม่รู้จักวิธีการขอและไม่สามารถรองรับกรัพยากรใดๆ ได้

#### 502 Bad Gateway

ข้อผิดพลาด 502 Bad Gateway หมายความว่าเว็บเซิร์ฟเวอร์ที่คุณเชื่อมต่อทำหน้าที่เป็นพื้นที่สำหรับการลั่งต่อข้อมูลจากเซิร์ฟเวอร์อื่น แต่ได้รับการตอบสนอง (Response) กิ้งไม่ดีจากเซิร์ฟเวอร์อื่นนั้น

## 503 Service Unavailable

บอกว่า Server ใช้การไม่ได้ หรือไม่ว่างที่จะรับและดำเนินการ Request โดยส่วนใหญ่แล้ว Server อยู่ในช่วงบำรุงรักษา

## 504 Gateway Timeout

เป็นรหัสการตอบสนองสถานะที่ส่งสัญญาณถึงความล้มเหลวของเซิร์ฟเวอร์ระหว่างการประมวลผลคำขอโดยปกติแล้วจะเป็นสัญญาณว่าคอมพิวเตอร์ที่กำหนดน้ำหนักเป็นพร็อกซี (เกตเวย์) ระหว่างคุณกับเซิร์ฟเวอร์ที่คุณเข้าถึงไม่ตอบสนองในเวลาที่เหมาะสม เป็นปัญหาที่พบบ่อยกับเซิร์ฟเวอร์การเข้าชมที่มีผู้เข้าชมเป็นจำนวนมาก

## 505 HTTP Version Not Supported

เซิร์ฟเวอร์ไม่รองรับหรือปฏิเสธที่จะรองรับเวอร์ชันprotoคือ HTTP ที่ใช้ในข้อความคำขอ เซิร์ฟเวอร์แจ้งว่าไม่สามารถทำตามคำขอโดยใช้เวอร์ชันหลักเดียวกันกับไดลอจินต์

## 508 Resource Limit Is Reached

ถึงขีดจำกัดของกรัพยากรที่กำหนดโดยโฮสต์เว็บ (เว็บเซิร์ฟเวอร์)

## 511 Network Authentication Required

เครื่อข่ายที่คุณพยายามใช้ต้องมีการตรวจสอบลิขิตร่างรูปแบบก่อนที่จะส่งคำขอไปยังเซิร์ฟเวอร์ ตัวอย่างเช่น การยอมรับข้อกำหนดและเงื่อนไขของ Wi-Fi หอตสปอต สามารถแก้ไขการเชื่อมต่อ

## 521 Web server is down Error

เป็นข้อความแสดงข้อผิดพลาดเฉพาะ Cloudflare (ผู้ให้บริการระบบเครือข่าย network ที่ใหญ่ที่สุดในโลก ซึ่งเป็นองค์การไม่แสวงหาผลกำไร) หมายความว่าเว็บเบราว์เซอร์ของคุณสามารถเชื่อมต่อกับ Cloudflare ได้สำเร็จ แต่ Cloudflare ไม่สามารถเชื่อมต่อกับเว็บเซิร์ฟเวอร์ต้นทางได้