

# Yii2

Database and Report



# Yii 2 Database and Report

พัฒนา Web Application ด้วย Yii 2: ฐานข้อมูล และรายงาน

โค้ชเอก

หนังสือเล่มนี้จำหน่ายที่ <http://www.codingthailand.com/yii2ebook>

เวอร์ชัน 1 ออกจำหน่ายวันที่ 1 มกราคม 2559

หนังสือเล่มนี้ผู้เขียนตั้งใจจัดทำขึ้นเพื่ออยากให้มีหนังสือภาษาไทยซักเล่มเกี่ยวกับ Yii Framework 2 ที่เน้นเนื้อหาเจาะลึกด้านฐานข้อมูล และการทำรายงานในรูปแบบต่างๆ ซึ่งคิดว่าผู้อ่านที่ต้องการพัฒนา Web Application จำเป็นต้องมีความรู้ ความเข้าใจ และสามารถนำไปใช้กับการทำงาน และใช้เพื่อพัฒนาองค์กรของตนเอง หวังว่าหนังสือเล่มนี้จะเป็นประโยชน์ ประหยัดเวลาการเรียนรู้ ขอขอบคุณทุกคนนะครับ ขอให้มีความสุข และสนุกกับการเรียนรู้ครับ

“ความสำเร็จไม่มีทางลัด ต่อให้เก่งแค่ไหน แต่ขาดวินัยกับความต่อเนื่อง  
คงยากที่จะผ่านอุปสรรคไปได้”



©2015 - 2016 โค้ชเอก

# สารบัญ

## บทที่ 1: เริ่มต้น และติดตั้ง Yii 2

- ทำไมต้องใช้ PHP Framework
- ทำความรู้จักกับ Yii Framework
- MVC และ Best Practices
- การติดตั้งต้องเตรียมอะไรบ้าง
- Extension ของ PHP ที่ควรเปิดไว้
- แนะนำ Advance Project Template
- แนะนำ Advanced Project Template ของ Nenad Zivkovic
- ขั้นตอนการติดตั้ง Advanced Project Template ผ่าน Composer
- แนะนำการใช้งาน Advanced Project Template เบื้องต้น
- โครงสร้างของ Advanced Project Template
- สร้างโปรเจกต์ใหม่ด้วย Netbeans
- การตั้งค่า Advanced Project Template ของ Nenad Zivkovic
- การตั้งค่า Time Zone เพื่อจัดรูปแบบวันที่ และเวลาในประเทศไทย
- การตั้งค่า และเปลี่ยนหน้าตาเว็บไซต์ (Theming)
- การติดตั้ง Admin Template (AdminLTE)
- การตั้งค่า AdminLTE
- การสร้างไฟล์เดอร์ Theme สำหรับ AdminLTE
- การเพิ่มเมนูด้านซ้ายให้กับ AdminLTE
- เทคนิคการใส่ Box components ของ AdminLTE ให้กับ GridView และ Form

## บทที่ 2: Yii2 กับการใช้งานด้านฐานข้อมูล

- การใช้งาน Database Migration
- การสร้างตารางใหม่โดยใช้ Database Migration
- การ Reverting หรือ Undo โดยใช้ Database Migration
- การใช้งาน Data Access Object
- Data Access Object: สร้างการติดต่อกับฐานข้อมูล
- Data Access Object: การส่งคำสั่ง SQL
- Data Access Object: เมธอดสำหรับ query ข้อมูล

- Data Access Object: ลองเขียน DAO เพื่อเรียกดู และแสดงข้อมูลจากตาราง position
- Data Access Object: ลองเขียน DAO เพื่อลบจากตาราง position
- Data Access Object: DAO สำหรับการเพิ่ม และแก้ไขข้อมูล
- Data Access Object: การใช้เทคนิค prepare เพื่อให้การเรียกดูข้อมูลมีประสิทธิภาพมากขึ้น
- Data Access Object: การป้องกันไม่ให้ชื่อคอลัมน์ในตาราง และชื่อตารางซ้ำกับคำสั่งของฐานข้อมูล
- ทำความรู้จักกับ Query Builder
- ขั้นตอนการใช้ Query Builder
- ฝึกเขียน Query Builder และ ดูรายละเอียด Method สำหรับการคิวรี่ข้อมูล
- การใช้งาน Active Record
- การสร้าง Models ด้วย Gii เพื่อใช้งานกับ Active Record
- ขั้นตอนการคิวรี่ข้อมูลด้วย Active Record
- การคิวรี่ข้อมูลด้วย Active Record แบบวิธีลัด (shortcut methods)
- การเข้าถึง Attributes ข้อมูล เมื่อใช้ Active Record
- การสร้าง Attributes ใหม่ อาจใช้คำนวณ หรือจัดรูปแบบข้อมูล เมื่อใช้ Active Record
- เทคนิคการใช้ asArray() กับ Active Record
- การเพิ่ม หรือแก้ไขข้อมูลด้วย Active Record
- การเพิ่มหรือลดค่าของคอลัมน์ (Updating Counters)
- การลบข้อมูล โดยใช้ Active Record
- การใช้ Transactions
- การ Join ตาราง โดยใช้ Active Record
- การใช้ Data Providers สำหรับแบ่งหน้า และจัดเรียงข้อมูล
- การใช้ SqlDataProvider สำหรับแบ่งหน้า และจัดเรียงข้อมูล
- การใช้งาน GridView
- การกรองข้อมูลใน GridView (filters)
- Workshop: การจัดการข้อมูลบุคลากร (Person)

### บทที่ 3: การสร้างรายงานด้วย Yii 2

- สร้างรายงานในรูปแบบ PDF และรองรับพอนต์ภาษาไทย ด้วย mPDF
- ขั้นตอนการติดตั้ง yii2-mpdf
- การส่งออกและนำเข้าไฟล์ Excel
- สร้างรายงานในรูปแบบ Chart ด้วย HighCharts

## บทที่ 4: โบนัสพิเศษ

- การตั้งค่าและทำงานกับอีเมล ด้วย SMTP
- ทดลองส่งเมล ด้วย SMTP
- การสร้าง email layout และ view สำหรับส่งเมล
- แนะนำการอัปเดต Yii2 เพื่อใช้งานจริง

## บทที่ 1: เริ่มต้น และติดตั้ง Yii 2

### ทำไมต้องใช้ PHP Framework

- มีการเขียนโค้ดที่เป็นมาตรฐาน ช่วยลดและกำจัดโค้ดที่ไม่จำเป็น
- ช่วยลดระยะเวลาในการทำงาน เช่น เรื่องความปลอดภัย การสร้างฟอร์ม เป็นต้น
- ช่วยทำให้การทำงานเป็นทีมง่ายขึ้น เพราะต้องเขียนโค้ดเป็นมาตรฐานเดียวกัน
- ช่วยในการบำรุงรักษาโค้ดได้ง่ายขึ้น
- มี community ที่เข้มแข็ง เราสามารถถาม และคอยขอคำแนะนำได้

### ทำความรู้จักกับ Yii Framework

Yii (อ่านออกเสียงว่า ยี่) เป็น PHP Framework ที่มีประสิทธิภาพการทำงานสูง มีคุณลักษณะเด่นหลายด้านที่ช่วยให้เราเขียนโปรแกรมได้ง่ายขึ้น ไม่ว่าจะเป็น MVC, DAO, Form input และการตรวจสอบความถูกต้องของข้อมูล (validation), AJAX, Security เป็นต้น นอกจากนี้ Yii ยังสามารถสร้าง Web Application ได้ทุกขนาด ไม่ว่าจะเป็นระบบ CMS, e-commerce, Restful Web service และยังให้นักพัฒนาต่อยอดเขียน extensions เสริมได้อีกด้วย

### MVC และ Best Practices

Yii มีรูปแบบการเขียนแบบ MVC (Model, View, Controller) ซึ่งการจะเขียนให้ดีขึ้น ต้องศึกษาแนวทางกันก่อน สรุปให้ดังนี้

#### สรุปการเขียน Model ที่ดี

- ประกอบด้วย โค้ดในส่วน business data
- ประกอบด้วย โค้ดในการส่วนของการตรวจสอบความถูกต้องของข้อมูล
- ประกอบด้วย method การทำงานในส่วนของ business logic
- อย่าเขียนโค้ดเกี่ยวกับการ request, session หรือโค้ดเกี่ยวกับสภาพแวดล้อมของระบบ
- ระวังหรือหลีกเลี่ยงเขียนโค้ดเกี่ยวกับ HTML ในส่วนของการแสดงผลใน Model ให้ไปเขียนที่ view แทน

#### สรุปการเขียน View ที่ดี

- View จะต้องมีโค้ดเฉพาะ HTML และ PHP ที่เกี่ยวข้องกับการแสดงผล จัดรูปแบบข้อมูลต่างๆเท่านั้น
- จะต้องไม่โค้ดเกี่ยวกับการ query ฐานข้อมูลต่างๆ
- หลีกเลี่ยงการรับค่า \$\_GET, \$\_POST เพราะเป็นหน้าที่ของ Controller
- ถ้าเรารับค่ามาจาก model จะต้องไม่ไปแก้ไขค่าที่รับมา

- ใช้คลาสในกลุ่ม Helper เพื่อช่วยในการจัดรูปแบบข้อมูล

สรุปการเขียน Controller ที่ดี

- มีไว้เขียนเกี่ยวกับ request ข้อมูล เช่น get, post, put เป็นต้น
- มีไว้เรียกเมธอดเกี่ยวกับ Models และ เรียก component ต่างๆ
- มีไว้ส่งข้อมูลต่างๆ ไปให้ views เพื่อนำไปแสดงผล
- ไม่ควรมีได้ผลการประมวลผลของ Models ถ้ามีให้ไปเขียนที่ Models ดีกว่า
- หลีกเลี่ยงการเขียน HTML และโค้ดที่เกี่ยวข้องกับการแสดงผลข้อมูล ให้ไปเขียนที่ view ดีกว่า

## การติดตั้งต้องเตรียมอะไรบ้าง

เนื่องจากหนังสือเล่มนี้ไม่ใช่หนังสือพื้นฐาน การเตรียมตัวอย่างแรกในการอ่านหนังสือเล่มนี้คือ เราจะต้องมีพื้นฐานภาษา PHP และ มีความรู้เกี่ยวกับการเขียนโปรแกรมในรูปแบบของ Object Oriented Programming หรือ OOP มาก่อน เพื่อให้เกิดประโยชน์สูงสุดในการเรียนรู้ สำหรับคนที่ยังไม่มีพื้นฐานความรู้ดังที่กล่าวมา ผมว่าแนะนำควรให้ศึกษาก่อนดีกว่าครับ

ในการติดตั้ง Yii 2 สิ่งที่เราต้องเตรียม ประกอบด้วย

1. XAMPP สำหรับจำลองเครื่องเราให้เป็น Web Server ประกอบด้วย Apache, PHP, MySQL/MariaDB และ phpMyAdmin
2. Netbeans สำหรับใช้เขียนโค้ด หรือจะใช้ IDE ที่ไหนก็ได้
3. Composer สำหรับการจัดการกับ PHP Packages หรือ Library ต่างๆ

ขั้นตอนการติดตั้งทั้งหมด สามารถเปิดดูวิดีโอได้ที่ [https://www.youtube.com/watch?v=6DH\\_aEaJ3X4](https://www.youtube.com/watch?v=6DH_aEaJ3X4)

## Extension ของ PHP ที่ควรเปิดไว้

บางครั้งระหว่างติดตั้ง Composer หรือ พัฒนา Web Application อาจมี errors สำหรับบางคำสั่ง ก่อนติดตั้ง Yii2 ควรไปตรวจสอบ หรือ เปิด extension ในไฟล์ php.ini ให้เรียบร้อย คือ ให้เปิดไฟล์ php.ini (C:\xampp\php\php.ini) ค้นหา extension แล้วเอาเครื่องหมาย ; ออก ข้างหน้า เสร็จแล้วบันทึกไฟล์แล้ว restart Apache ส่วนรายการ extension ที่ควรเปิดมีดังต่อไปนี้

extension=php\_bz2.dll    extension=php\_curl.dll    extension=php\_mbstring.dll    extension=php\_fileinfo.dll

extension=php\_gd2.dll    extension=php\_openssl.dll    extension=php\_intl.dll

```
990 extension=php_bz2.dll
991 extension=php_curl.dll
992 extension=php_mbstring.dll
993 extension=php_exif.dll
994 extension=php_fileinfo.dll
995 extension=php_gd2.dll
996 extension=php_gettext.dll
997 ;extension=php_gmp.dll
998 extension=php_intl.dll
999 extension=php_openssl.dll
```

## แนะนำ Advance Project Template

หลังจากติดตั้งเครื่องมือต่างๆ เรียบร้อยแล้ว เราจะมาติดตั้ง Yii 2 กัน ซึ่งโดยปกติรูปแบบของโปรเจกต์ของ Yii 2 จะมีอยู่ 2 รูปแบบด้วยกัน ได้แก่ Basic Project Template และ Advanced Project Template ในหนังสือเล่มนี้ผมจะเลือกใช้รูปแบบ Advanced Project Template เพราะมีการแยกส่วน Front-end และ Back-end กันอย่างชัดเจน และมีตัวช่วยอื่นๆ เช่น มีหน้าเว็บลงทะเบียนผู้ใช้ มีหน้าและฟังก์ชันการกู้คืนรหัสผ่านผู้ใช้ โดยที่เราไม่ต้องออกแรงเพิ่ม 😊 แต่!! ยังมีดีกว่านั้นคือ ในหนังสือเล่มนี้เราจะใช้ Advanced Template ที่จะช่วยทุ่นแรงเราขึ้นไปอีก โดยมีคนที่ทำ Advanced Project Template สำเร็จรูปที่มีหลากหลายฟังก์ชัน และเราสามารถติดตั้งมาใช้ได้เลย

## แนะนำ Advanced Project Template ของ Nenad Zivkovic

เพื่อความเร็ว และประหยัดเวลาในการพัฒนา Web Application หนังสือเล่มนี้จะใช้ Advanced Project Template ของ Nenad Zivkovic ให้เราเข้าไปดูรายละเอียดต่างๆได้ที่ <https://github.com/nenad-zivkovic/yii2-advanced-template> หรือใครจะใช้ Advance Project Template แบบมาตรฐานที่มากับ Yii ก็ได้เช่นเดียวกัน

สำหรับ Advanced Project Template ของ Nenad Zivkovic นั้นเค้าได้เขียนเพิ่มฟังก์ชันต่างๆจาก Advanced Project Template แบบมาตรฐานของ Yii หลายส่วนด้วยกัน ได้แก่

- มีระบบลงทะเบียนผู้ใช้ (User) ให้ พร้อมกำหนด account activation ได้ด้วย (กำหนดได้ว่าผู้ใช้งานต้องยืนยันตัวตนผ่านอีเมลหลังลงทะเบียนหรือไม่) โดยกำหนดได้ที่ common/config/params.php
- สามารถล็อกอินได้ทั้งใช้แบบ username/password แบบปกติ และยังล็อกอินได้แบบ email/password ได้ด้วย แนนอนเรา สามารถตั้งค่าระบบได้ที่ โดยกำหนดได้ที่ common/config/params.php เช่นเดียวกัน
- มีระบบการจัดการผู้ใช้ หรือ RBAC มาให้เรียบร้อย ทำให้เราสามารถจัดการในส่วนของผู้ใช้ได้เลย และสามารถกำหนดบทบาท หรือสิทธิ์ในการเข้าถึงหน้าเว็บได้ง่ายขึ้น
- มีการเก็บ Session ลงในฐานข้อมูล
- มีระบบธีม (Theming) สำหรับเปลี่ยนหน้าเว็บไซต์ได้หลากหลายรูปแบบ และยังสามารถเพิ่มธีมได้ง่าย
- มีระบบเว็บไซต์แบบหลายภาษา (Translation) ทำให้เราสามารถกำหนดให้เว็บไซต์มีการแสดงผลแบบหลายภาษาได้
- มีระบบ และหน้าเว็บให้ admin หรือ The Creator (ตัวเรา) จัดการกับผู้ใช้ได้เลย
- มีระบบตรวจสอบรหัสผ่านสำหรับแนะนำผู้ใช้งานว่ารอกยาก-ง่ายเกินไปหรือไม่

และแนนอนเรายังใช้ความสามารถของ Advanced Project Template แบบมาตรฐานของ Yii ได้ครบถ้วน เตรียมอินเตอร์เน็ตให้พร้อม แล้วมาติดตั้งกันเลยครับ!



## ขั้นตอนการติดตั้ง Advanced Project Template ผ่าน Composer

1. เข้าไปที่โฟลเดอร์ `htdocs` ของ XAMPP (ปกติจะอยู่ที่ `C:\xampp\htdocs`) แล้วคลิกขวา เลือก `Use Composer here` เราจะได้หน้าต่าง `command line` ขึ้นมาเพื่อพร้อมพิมพ์คำสั่งสำหรับติดตั้ง หรือถ้าไม่คลิกขวาก็สามารถใช้คำสั่ง `command line` ได้โดย `cd` เข้าไปยังโฟลเดอร์ `htdocs` ก็ได้เหมือนกัน เช่น `cd c:\xampp\htdocs`



```
C:\Windows\System32\cmd.exe

Basic usage: composer <command>
For more information just type "composer".

C:\xampp\htdocs>
```

**Note:** สำหรับคนที่ใช้ Linux หรือ Mac OS X ให้ติดตั้ง Composer โดยพิมพ์คำสั่ง:

```
curl -sS https://getcomposer.org/installer | php
```

```
mv composer.phar /usr/local/bin/composer
```

2. พิมพ์คำสั่ง `composer self-update` เพื่อ update เวอร์ชันของ Composer ให้ล่าสุด แล้วกดปุ่ม `Enter` รอสักครู่นะขึ้นคำว่า `Updating to version...` ก็เป็นอันเสร็จเรียบร้อย



```
C:\Windows\System32\cmd.exe

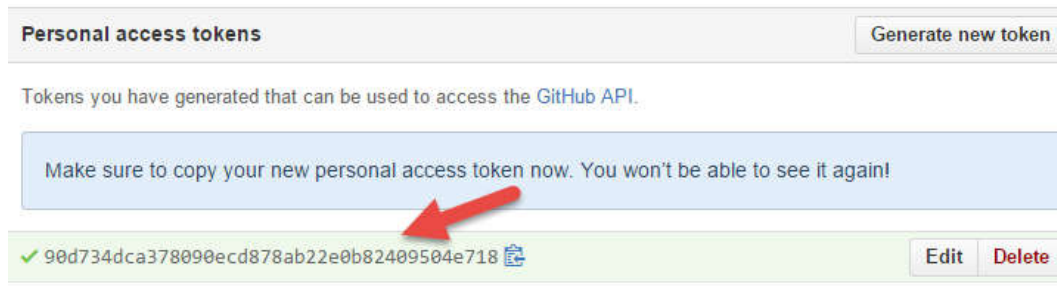
Basic usage: composer <command>
For more information just type "composer".

C:\xampp\htdocs>composer self-update
Updating to version f31799b739265e7cb1797b395050ffcd57671335.
  Downloading: 100%
Use composer self-update --rollback to return to version d7d91269ec4fb0da171a1c38fa1a47c493627dc0

C:\xampp\htdocs>
```

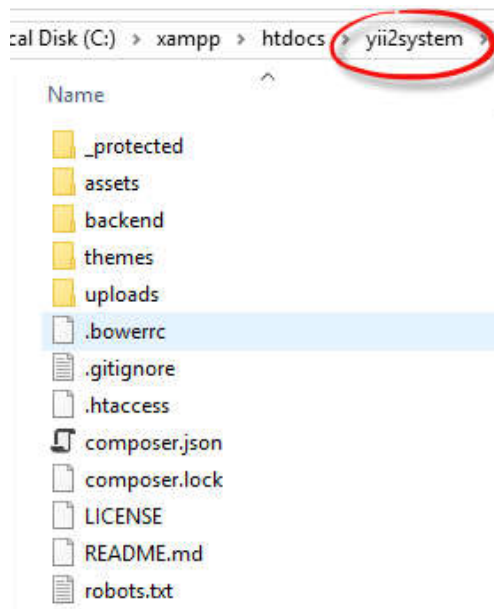
3. ติดตั้ง composer asset plugin พิมพ์คำสั่ง `composer global require "fxp/composer-asset-plugin:~1.1.1"` แล้วกดปุ่ม Enter รอจนกว่าจะเรียบร้อยเหมือนเดิม

**Note:** ระหว่างขั้นตอนที่ 3 สำหรับคนที่เพิ่งเคยติดตั้งครั้งแรก จะมีการถามหา token ของ GitHub ดังนั้น ให้ไปสร้าง token ก่อนโดยไปสมัครสมาชิกได้ที่เว็บ [github.com](https://github.com) จากนั้นล็อกอินแล้วเข้าไปที่ <https://github.com/settings/tokens> คลิกปุ่ม Generate new token กรอก Token description คลิกปุ่ม Generate token แล้วใช้เมาส์ลากคลุม token เพื่อคัดลอก (Copy) แล้วนำไปวาง (Paste) ที่หน้าจอก command line ได้เลย



4. สร้างโปรเจกใหม่โดยเราจะใช้ Advanced Project Template ของ Nenad Zivkovic พิมพ์คำสั่ง `composer create-project nenad/yii2-advanced-template yii2system` แล้วกดปุ่ม Enter รอจนเสร็จเรียบร้อย แล้วให้ลองตรวจสอบดูว่ามีโฟลเดอร์ yii2system ถูกสร้างขึ้นในโฟลเดอร์ htdocs หรือไม่ ถ้ามี แสดงว่าติดตั้งเรียบร้อยแล้ว

**Note:** ตรงคำสั่ง yii2system คือ ชื่อโฟลเดอร์ที่จะถูกสร้างที่ C:\xampp\htdocs สำหรับเก็บไฟล์โปรเจกของเรานั่นเอง



5. พิมพ์คำสั่งเพื่อเข้าไปในโฟลเดอร์ \_protected ดังนี้ `cd yii2system/_protected/` แล้วกดปุ่ม Enter (อย่าลืมว่า yii2system คือชื่อโปรเจกของเรา ถ้าเปลี่ยนชื่อเป็นอย่างอื่นก็อย่าลืมเปลี่ยนตรงนี้ด้วย) ต่อด้วยพิมพ์คำสั่ง `php init` กด Enter พิมพ์ `0` แล้ว Enter อีกครั้งเพื่อเลือกสภาพแวดล้อมการพัฒนาโปรแกรม (Development) พิมพ์ `yes` แล้วกด Enter

```
C:\Windows\System32\cmd.exe

C:\xampp\htdocs\yii2system\_protected>php init 1
Yii Application Initialization Tool v1.0

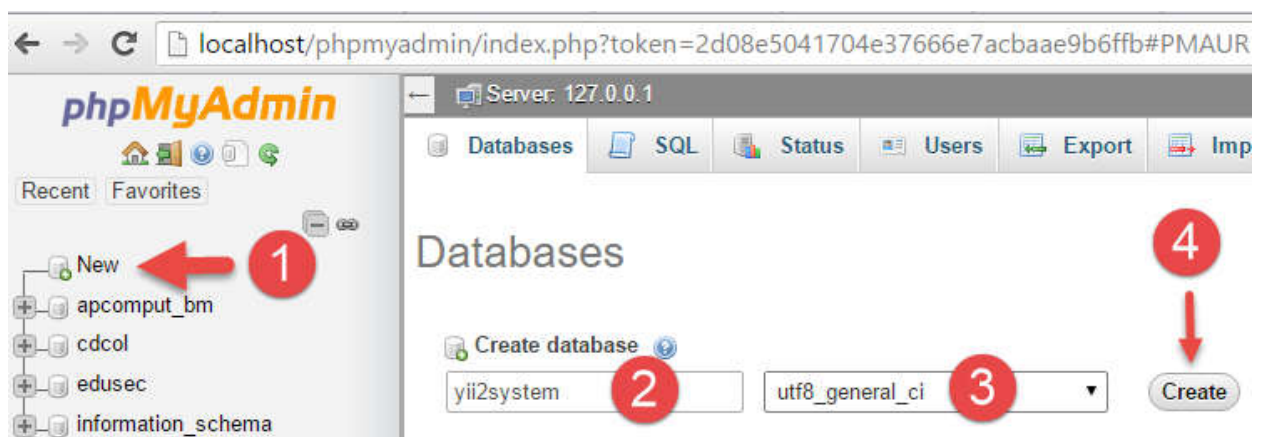
Which environment do you want the application to be initialized in?

[0] Development
[1] Production

Your choice [0-1, or "q" to quit] 0 2

Initialize the application under 'Development' environment? [yes|no] yes 3
Start initialization ...
generate backend/index-test.php
generate backend/index.php
generate index-test.php
generate index.php
generate _protected/backend/config/main-local.php
generate _protected/backend/config/params-local.php
generate _protected/common/config/main-local.php
generate _protected/common/config/params-local.php
generate _protected/console/config/main-local.php
generate _protected/console/config/params-local.php
generate _protected/frontend/config/main-local.php
generate _protected/frontend/config/params-local.php
generate _protected/yii
generate cookie validation key in _protected/backend/config/main-local.php
generate cookie validation key in _protected/frontend/config/main-local.php
```

6. สร้างฐานข้อมูล (database) โดยใช้ phpMyAdmin เปิด Browser แล้วพิมพ์ <http://localhost/phpmyadmin> คลิกที่ ลิงก์ New เพื่อสร้างฐานข้อมูลใหม่ จากนั้นกรอกชื่อฐานข้อมูล รวมทั้ง Collation เสร็จแล้วคลิกปุ่ม Create



7. เปิดไฟล์ main-local.php เพื่อตั้งค่าฐานข้อมูลให้ถูกต้อง yii2system /\_protected/common/config/main-local.php แล้วแก้ไขไฟล์ให้ตรงกับฐานข้อมูลที่ได้สร้างไว้ ได้แก่ dbname, username, password

```
<?php
return [
    'components' => [
        'db' => [
            'class' => 'yii\db\Connection',
            'dsn' => 'mysql:host=localhost;dbname=yii2system',
            'username' => 'root',
            'password' => '',
            'charset' => 'utf8',
        ],
        'mailer' => [
            'class' => 'yii\swiftmailer\Mailer',
            'viewPath' => '@common/mail',
            // send all mails to a file by default. You have to set
            // 'useFileTransport' to false and configure a transport
            // for the mailer to send real emails.
            'useFileTransport' => true,
        ],
    ],
];
```

8. กลับมาที่หน้าต่าง command line (สังเกตว่าเรายังอยู่ในโฟลเดอร์ \_protected อยู่) พิมพ์ `yii migrate` แล้วกด Enter จากนั้นจะมีรายการตารางของฐานข้อมูลแสดงขึ้นมา พิมพ์ `yes` เพื่อสร้างตารางทุกตาราง โดยตารางต่าง ๆ นั้น ประกอบด้วยตารางผู้ใช้ ตารางสำหรับ rbac ตารางสำหรับเก็บ session และ ตารางตัวอย่างสำหรับเก็บบทความ (article)

Note: ถ้าใช้ Linux หรือ Mac OS ให้ใช้คำสั่ง `./yii migrate`

```
C:\Windows\System32\cmd.exe

C:\xampp\htdocs\yii2system\_protected>yii migrate 1
Yii Migration Tool (based on Yii v2.0.6)

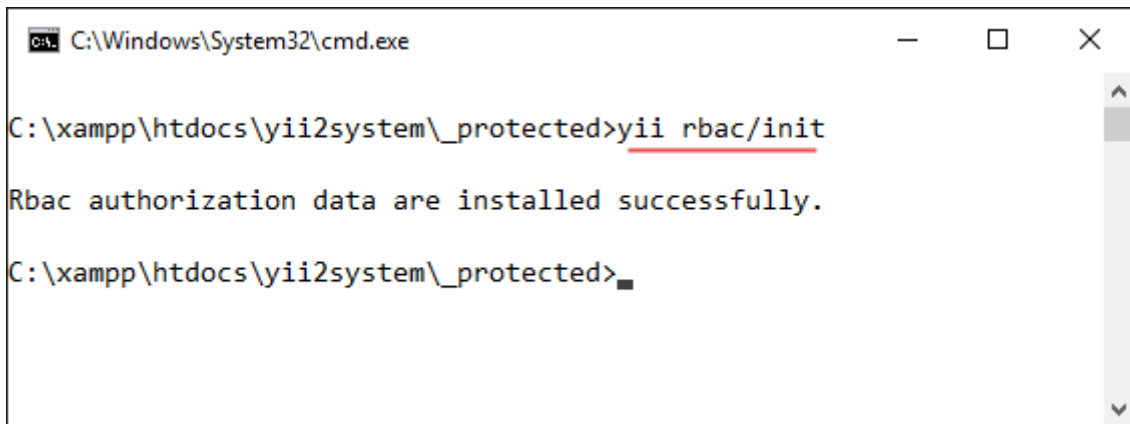
Creating migration history table "migration"...Done.
Total 4 new migrations to be applied:
    m141022_115823_create_user_table
    m141022_115912_create_rbac_tables
    m141022_115922_create_session_table
    m150104_153617_create_article_table

Apply the above migrations? (yes|no) [no]:yes 2
*** applying m141022_115823_create_user_table
> create table {%user%} ... done (time: 0.468s)
*** applied m141022_115823_create_user_table (time: 0.567s)

*** applying m141022_115912_create_rbac_tables
> create table {%auth_rule%} ... done (time: 0.415s)
> create table {%auth_item%} ... done (time: 0.406s)
> create index idx-auth_item-type on {%auth_item%} (type) ... done (time: 0.344s)
*** applied m141022_115912_create_rbac_tables (time: 1.816s)

*** applying m141022_115922_create_session_table
> create table {%session%} ... done (time: 0.411s)
*** applied m141022_115922_create_session_table (time: 0.506s)
```

9. พิมพ์คำสั่ง `yii rbac/init` แล้วกด Enter เพื่อสร้างระบบจัดการผู้ใช้แบบ rbac สร้างตารางบทบาท และสิทธิ์การใช้งานต่างๆ



```
C:\Windows\System32\cmd.exe

C:\xampp\htdocs\yii2system\_protected>yii rbac/init

Rbac authorization data are installed successfully.

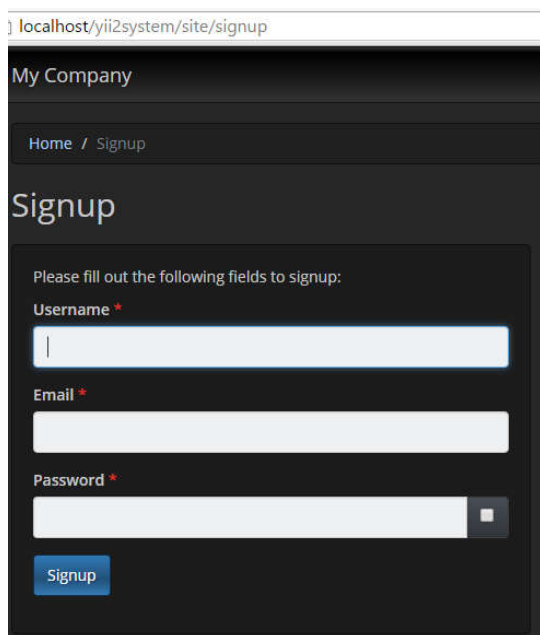
C:\xampp\htdocs\yii2system\_protected>
```

Note: ถ้าใช้ Linux หรือ Mac OS ให้ใช้คำสั่ง `./yii migrate/init`

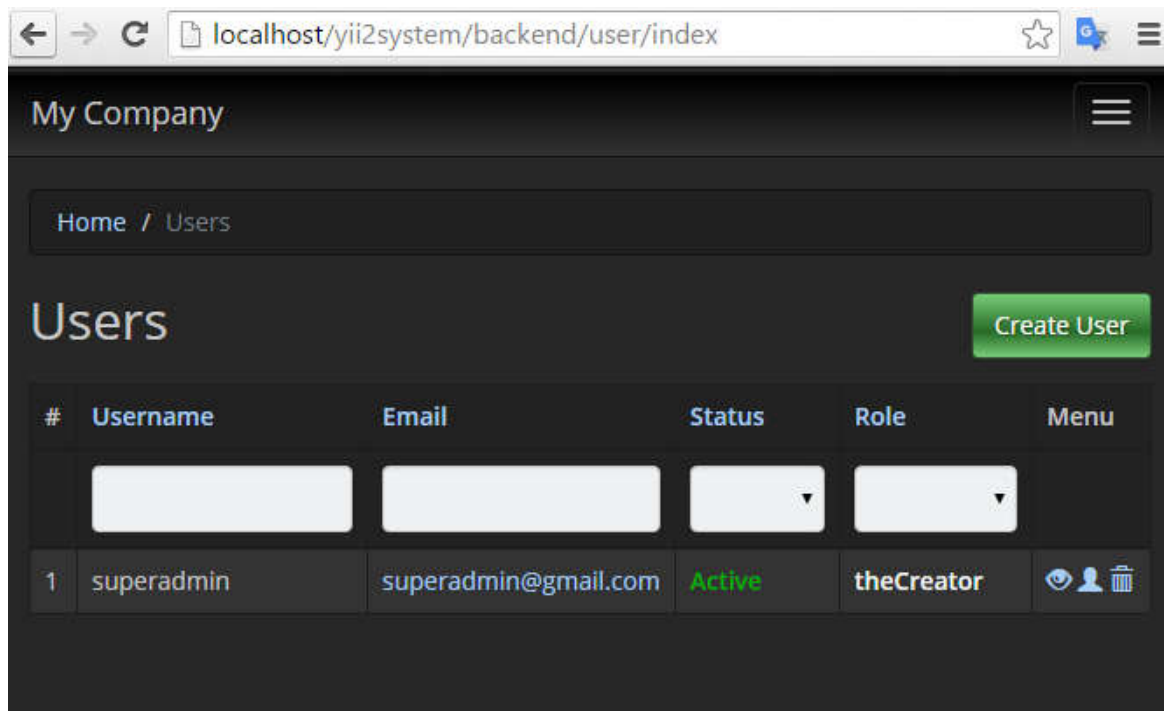
10. เสร็จเรียบร้อยแล้ว! เปิด Web browser แล้วลองพิมพ์ `http://localhost/yii2system` เพื่อเข้าไปในส่วนของ front-end และลองพิมพ์ `http://localhost/yii2system/backend` เพื่อเข้าไปส่วนของ back-end

## แนะนำการใช้งาน Advanced Project Template เบื้องต้น

หลังจากติดตั้ง Advanced Project Template เรียบร้อยแล้ว สิ่งที่ต้องทำต่อไป คือ แนะนำให้ลงทะเบียน (เมนู Signup) เพื่อสร้างผู้ใช้ใหม่ สำหรับผู้ใช้งานแรกที่ลงทะเบียนจะได้เป็น **ผู้ดูแลระบบสูงสุด** หรือเรียกบทบาทว่า *The Creator* สามารถทำอะไรก็ได้ใน Web Application ของเรา ส่วนการลงทะเบียนครั้งต่อไปจะได้บทบาทหรือสิทธิ์เป็น **สมาชิก** หรือ *member* นั่นเอง



เมื่อสร้างผู้ใช้สำหรับบทบาท The Creator (super admin) เสร็จแล้ว ให้ลองเข้าไปในส่วนของ back-end จะเห็นว่า The Creator สามารถจัดการในส่วนของผู้ใช้ได้ ไม่ว่าจะเป็นการเพิ่ม แก้ไข หรือลบ



## โครงสร้างของ Advanced Project Template

โครงสร้างโฟลเดอร์รูปแบบ Advanced Project Template ของ Nenad Zivkovic จริงๆ แล้วก็เหมือนกับโครงสร้าง Advanced Project Template แบบมาตรฐานของ Yii เพียงแต่จะมีการจัดกลุ่มโฟลเดอร์หลักต่างๆมาไว้ในโฟลเดอร์ \_protected นั่นเอง นอกจากนั้นยังมีการเพิ่มโฟลเดอร์โครงสร้างต่างๆเข้ามาด้วย เช่น themes เป็นต้น โครงสร้าง Advanced Project Template ของ Nenad Zivkovic มีดังต่อไปนี้

\_protected

backend

- assets/ contains backend assets definition
- config/ contains backend configurations
- controllers/ contains Web controller classes
- helpers/ contains helper classes
- models/ contains backend-specific model classes
- runtime/ contains files generated during runtime
- views/ contains view files for the Web application

common

- config/ contains shared configurations
- mail/ contains view files for e-mails

models/	contains model classes used in both backend and frontend
rbac/	contains role based access control classes
console	
config/	contains console configurations
controllers/	contains console controllers (commands)
migrations/	contains database migrations
models/	contains console-specific model classes
runtime/	contains files generated during runtime
environments	contains environment-based overrides
frontend	
assets/	contains frontend assets definition
config/	contains frontend configurations
controllers/	contains Web controller classes
models/	contains frontend-specific model classes
runtime/	contains files generated during runtime
views/	contains view files for the Web application
widgets/	contains frontend widgets
assets	contains application assets generated during runtime
backend	contains the entry script and Web resources for backend side of application
themes	contains frontend themes
uploads	contains various files that can be used by both frontend and backend applications

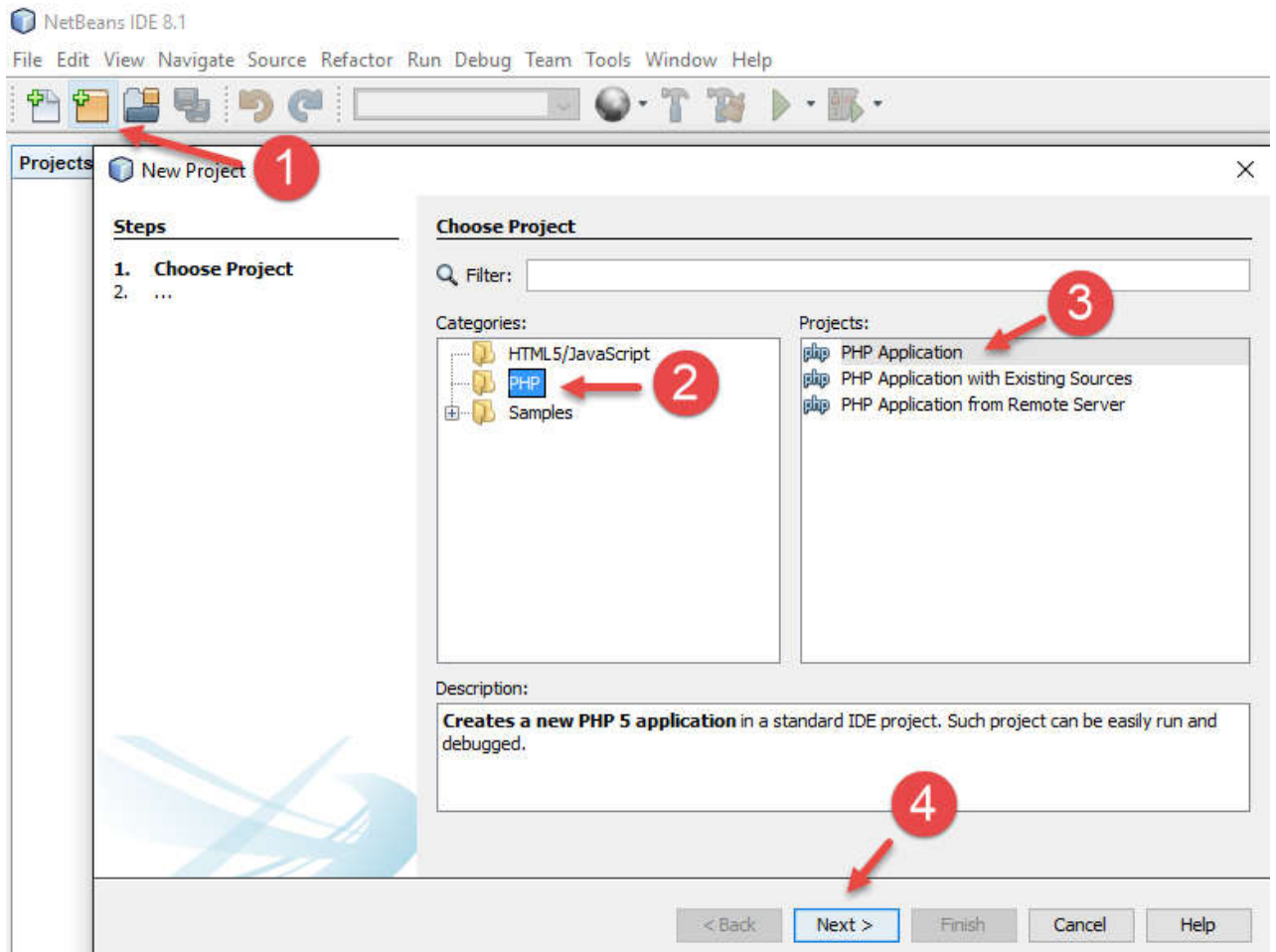
ในที่นี้จะไม่ได้แปลความหมายของไฟล์เดอร์ทั้งหมด เพราะค่อนข้างมีความหมายตรงตัวอยู่แล้ว ส่วนไฟล์เดอร์สำคัญๆ จะพูดถึงในหัวข้อต่อไปครับ

**Note:** ในการอ้างอิงที่อยู่ไฟล์ และไฟล์เดอร์ในหัวข้อต่อไป บางครั้งจะไม่มีการอ้างอิงชื่อไฟล์เดอร์ `_protected` อีก เช่น ถ้าเขียนว่า `common/config/main.php` ให้หมายถึง ไฟล์ `common` นั้นอยู่ในไฟล์เดอร์ `_protected` นั่นที่

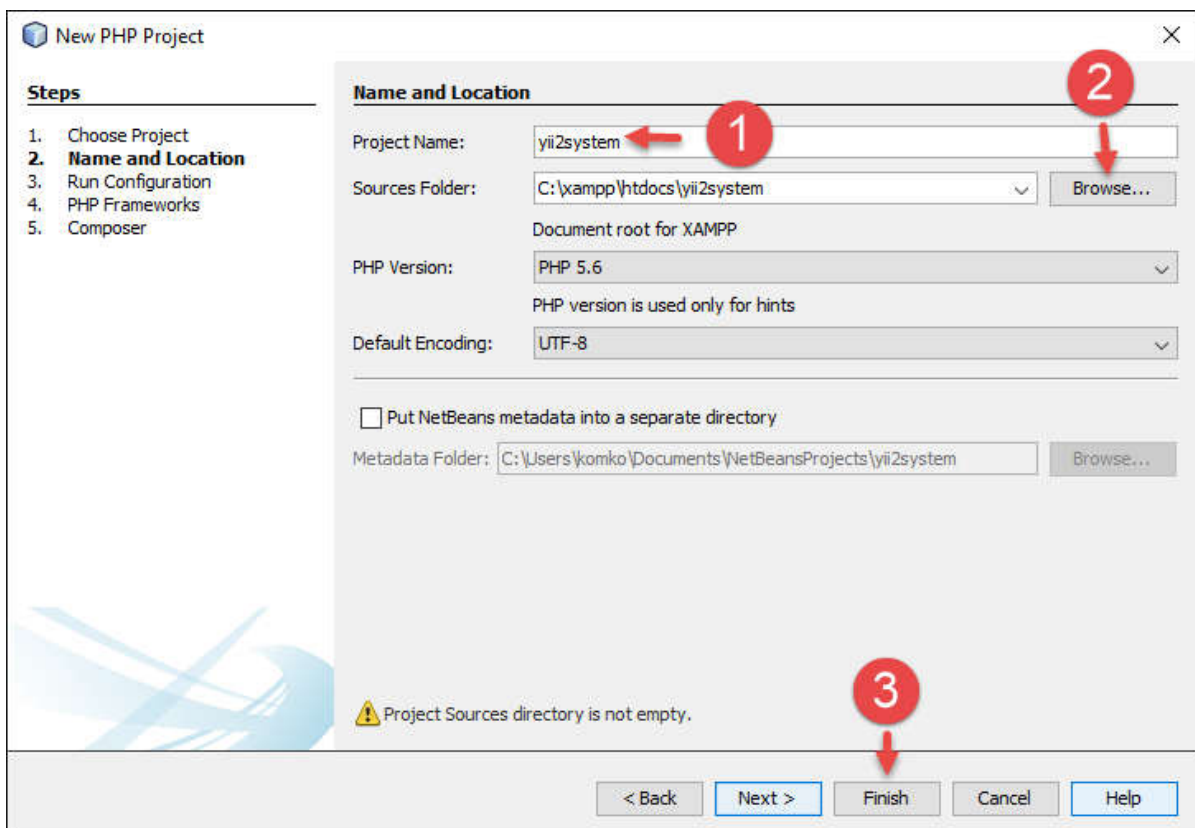
## สร้างโปรเจกต์ใหม่ด้วย Netbeans

เมื่อเปิดโปรแกรม Netbeans ขึ้นมาแล้ว แนะนำให้สร้างโปรเจกต์ใหม่ โดยมีขั้นตอน ดังนี้

1.คลิกไอคอน New Project -> เลือก PHP -> PHP Application -> กดปุ่ม Next



2. ตั้งชื่อ Project Name -> กดปุ่ม Browse.. เพื่อเลือกโฟลเดอร์ yii2system -> กดปุ่ม Finish เป็นอันเสร็จเรียบร้อย



เพียงเท่านี้ก็พร้อมเขียนโค้ดกันได้แล้ว!



## การตั้งค่า Advanced Project Template ของ Nenad Zivkovic

ถ้าเราใช้ Advanced Project Template ของ Nenad Zivkovic สามารถตั้งค่าระบบเบื้องต้นได้เลย ที่ไฟล์ common/config/params.php

```
<?php
```

```
return [
```

```
//-----//
```

```
// SYSTEM SETTINGS
```

```
//-----//
```

```
/**
```

```
 * Registration Needs Activation.
```

```
 *
```

```
 * ถ้าตั้งค่าเป็น true ตอนลงทะเบียนเสร็จแล้วผู้ใช้จะต้อง activate หรือยืนยันตัวตนผ่านทางอีเมล
```

```
 */
```

```
'rna' => false,
```

```
/**
```

```
 * Login With Email.
```

```
 *
```

```
 * ถ้าตั้งค่าเป็น true สามารถล็อกอินแบบผสมได้ทั้ง username และ email
```

```
 */
```

```
'lwe' => false,
```

```
/**
```

```
 * Force Strong Password.
```

```
 *
```

```
 * ถ้าตั้งค่าเป็น true สามารถใช้ StrengthValidator เพื่อตรวจสอบความปลอดภัยของรหัสผ่านได้
```

```
 */
```

```
'fsp' => false,
```

```

/**
 * Set the password reset token expiration time.
 */

'user.passwordResetTokenExpire' => 3600,

//-----//

// EMAILS

//-----//

/**
 * Email used in contact form.
 * Users will send you emails to this address.
 * ตั้งค่าอีเมลของระบบสำหรับ admin
 */

'adminEmail' => 'codingthailand@gmail.com',

/**
 * Not used in template.
 * You can set support email here.
 * ตั้งค่าอีเมลของระบบสำหรับ support
 */

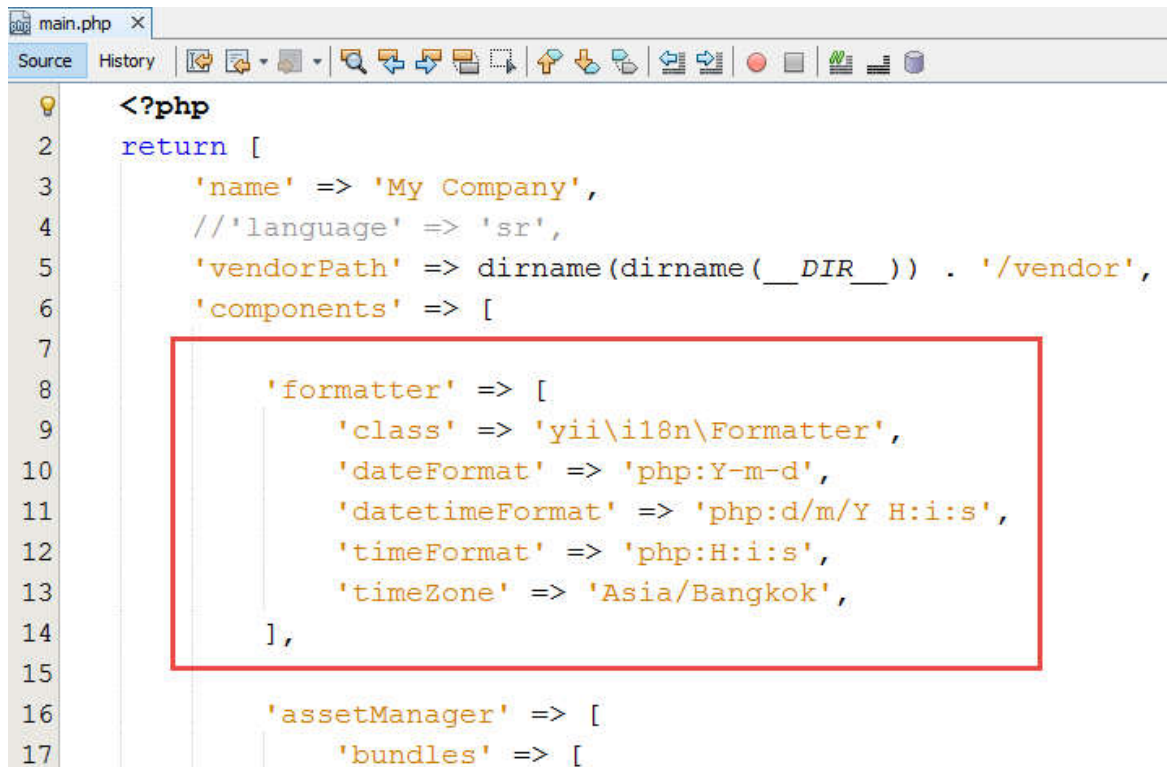
'supportEmail' => 'codingthailand@gmail.com',

];

```

## การตั้งค่า Time Zone เพื่อจัดรูปแบบวันที่ และเวลาในประเทศไทย

การตั้งค่าวันที่และเวลาควรเริ่มทำก่อนที่จะสร้าง Web Application เพราะถ้าวันที่และเวลาไม่ตรงอาจมีปัญหากับการจัดการข้อมูลตรงนี้ได้  
 ในอนาคต การตั้งค่าให้เปิดไฟล์ common/config/main.php ถ้าเราตั้งตรงนี้ (common) ก็จะสามารถใช้ได้ทั้งส่วน front-end และ back-  
 end โดยให้เพิ่มโค้ดเป็น array อีก 1 ชุด ได้ components และตั้งชื่อว่า formatter



```
<?php
return [
    'name' => 'My Company',
    //'language' => 'sr',
    'vendorPath' => dirname(dirname(__DIR__)) . '/vendor',
    'components' => [
        'formatter' => [
            'class' => 'yii\i18n\Formatter',
            'dateFormat' => 'php:Y-m-d',
            'datetimeFormat' => 'php:d/m/Y H:i:s',
            'timeFormat' => 'php:H:i:s',
            'timeZone' => 'Asia/Bangkok',
        ],
        'assetManager' => [
            'bundles' => [
```

หากเราสังเกตจะเห็นว่าใน Yii จะใช้คลาส Formatter ในการกำหนดและจัดรูปแบบของวันที่และเวลา ถ้าเริ่มต้นด้วย php: หมายถึงให้จัดรูปแบบการแสดงผลวันที่และเวลาตามรูปแบบของ PHP ดูรายละเอียดได้ที่ <http://php.net/manual/en/function.date.php> ส่วน timeZone ก็ให้ตั้งค่าเป็น Asia/Bangkok และเพื่อเป็นการทดสอบว่าการแสดงผลวันที่และเวลาถูกต้องให้ เปิดไฟล์ frontend/views/layouts/main.php จากนั้นลองแทรกโค้ด php เข้าไปในส่วนไหนก็ได้ของหน้าเว็บเพจหรืออาจเป็น footer ก็ได้ เมื่อแสดงผลแล้วให้เช็คด้วยว่าวันที่และเวลาตรงกับวันที่และเวลาเครื่องหรือไม่ ถ้าตรงแสดงว่าตั้งค่าได้ถูกต้องครับ

```
<?php echo Yii::$app->formatter->asDate(time()); ?>
```

```
<?php echo Yii::$app->formatter->asDatetime(time()); ?>
```

```
<?php echo Yii::$app->formatter->asTime(time()); ?>
```

**Note:** ดูรายละเอียดการจัดรูปแบบของคลาส Formatter ได้ที่ <http://www.yiiframework.com/doc-2.0/yii-i18n-formatter.html>

หากเกิด errors บางครั้งเราอาจต้องไปแก้ไข php.ini ให้มองหาบรรทัด `extension=php_intl.dll` เอาเครื่องหมาย ; ข้างหน้าออก เพื่อเปิด extension Internationalization เสร็จแล้วให้ restart Apache อีกครั้ง

## การตั้งค่า และเปลี่ยนหน้าตาเว็บไซต์ (Theming)

Advanced Project Template ของ Nenad Zivkovic สามารถเลือกธีมได้ 4 รูปแบบ ประกอบด้วย default, slate, spacelab และ cerulean การตั้งค่าเพื่อเลือกธีมนั้น ให้เปิดไฟล์ frontend/config/main.php โดยระบุชื่อ theme ในส่วนของ pathMap และ baseUrl

```
'components' => [
    // here you can set theme used for your frontend application
    // - template comes with: 'default', 'slate', 'spacelab' and 'cerulean'
    'view' => [
        'theme' => [
            'pathMap' => ['@app/views' => '@webroot/themes/cerulean/views'],
            'baseUrl' => '@web/themes/cerulean',
        ],
    ],
],
```

ในหนังสือเล่มนี้ ส่วน front-end จะใช้ theme ชื่อว่า cerulean เพราะสะดวกดี อย่าลืม save ไฟล์แล้วลองรีเฟรชเว็บดูนะครับ ในส่วนของ back-end นั้นจะพาดิตตั้ง theme ชื่อว่า AdminLTE หรือใครอยากใช้ theme ที่มีอยู่แล้วก็ได้เช่นเดียวกันแค่ไปตั้งค่าที่

backend/config/main.php

**Note:** สำหรับคนที่อยากเพิ่ม theme เองสามารถสร้างโฟลเดอร์ตั้งชื่อ theme ขึ้นมาโดยเลียนแบบของเดิมที่มีอยู่แล้ว จากนั้นก็ config ได้เลย เราสามารถแก้ไขไฟล์ theme/ชื่อtheme/css/site.css เพื่อปรับแต่งหน้าตาเว็บได้ แต่ถ้าอยากได้ theme ในแนวนี้อีกเพิ่ม แนะนำให้ไปโหลดไฟล์ bootstrap.min.css ได้ที่เว็บ <https://bootswatch.com> ครับ

## การติดตั้ง Admin Template (AdminLTE)

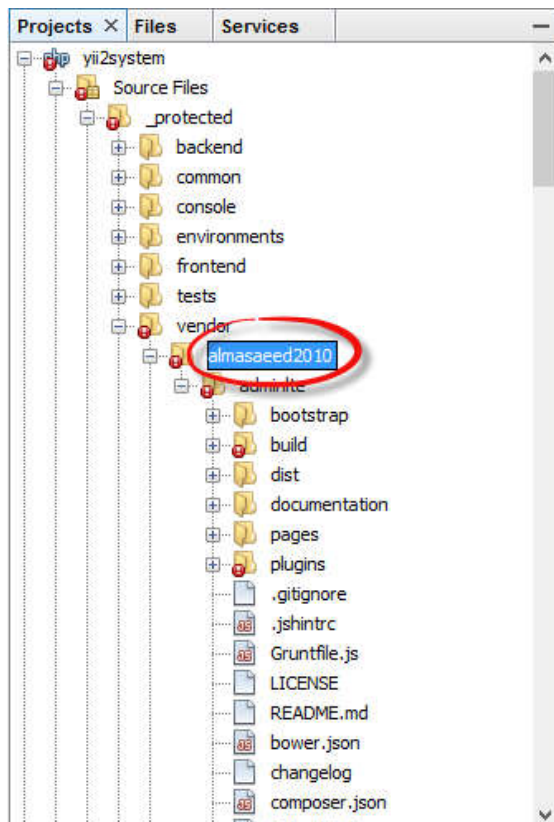
ปกติโดยทั่วไป Web Application จะมีระบบ back-end หรือเรียกง่ายๆว่า ระบบหลังบ้าน ซึ่งเป็นหน้าที่รวบรวมข้อมูลสรุปรายการสำคัญๆในระบบของเรา มีส่วนแสดงเมนูที่หลากหลาย รวมถึงสามารถแสดงรายงานต่างๆของระบบได้ และเพื่อเป็นการประหยัดเวลาในการพัฒนาในส่วนนี้จะขอแนะนำ admin template ที่ชื่อว่า “AdminLTE” ครับ ซึ่งรายละเอียดต้นฉบับ และคู่มือการใช้งานทั้งหมดจะอยู่ที่เว็บไซต์ <https://almsaeedstudio.com/>

ในส่วนการติดตั้ง AdminLTE เราจะใช้ตัวช่วยในการติดตั้ง ซึ่งมีคนทำไว้แล้วสำหรับ Yii2 ซึ่งง่ายและสะดวกต่อการใช้งานมาก เราไม่ต้องมาเสียเวลานำ js หรือ css เข้ามาเอง ดูรายละเอียดได้ที่เว็บ <https://github.com/dmstr/yii2-adminlte-asset> ขั้นตอนการติดตั้ง มีดังต่อไปนี้

1. เปิดไฟล์ composer.json แล้วพิมพ์ "dmstr/yii2-adminlte-asset": "2.\*" ในส่วนของ require (อย่าลืมเพิ่มเครื่องหมายคอมม่าบรรทัดสุดท้าย)



2. เข้าไปที่โฟลเดอร์โปรเจก C:\xampp\htdocs\yii2system คลิกขวาเลือก Use Composer here เพื่อเปิดหน้าต่าง command line จากนั้นพิมพ์ `composer update` แล้วกด Enter รอสักครู่จนกว่าการติดตั้งจะเรียบร้อย (ช้าเร็วขึ้นกับอินเทอร์เน็ตด้วย)
3. ตรวจสอบว่าไฟล์มาครบหรือไม่ โดยเข้าไปดูได้ที่โฟลเดอร์ `_protected/vendor/almasaeed2010/adminlte/` หากใน netbeans ไม่ขึ้นอาจลอง restart netbeans อีกครั้ง หรือเข้าไปดูที่ path โดยตรง



Note: หากติดตั้งตามขั้นตอนด้านบนไม่สำเร็จให้เปิด command line ขึ้นมาแต่เปลี่ยนจาก `composer update` เป็น `composer require dmstr/yii2-adminlte-asset "2.*"` แล้วกด Enter แทน

## การตั้งค่า AdminLTE

เมื่อติดตั้งเรียบร้อยแล้วต่อไปเป็นการตั้งค่า theme ในส่วนของ back-end ให้เปิดไฟล์ `backend/config/main.php` โดยต้องเพิ่ม 2 ส่วนด้วยกัน ได้แก่ 1. การเลือก Skins และ 2. การ pathMap และส่วนของ baseUrl

1. การเลือกหน้าตาหรือ skin ของ AdminLTE มีหลาย skin ให้เลือกดังต่อไปนี้

"skin-blue",  
"skin-black",  
"skin-red",  
"skin-yellow",  
"skin-purple",

"skin-green",  
"skin-blue-light",  
"skin-black-light",  
"skin-red-light",  
"skin-yellow-light",  
"skin-purple-light",  
"skin-green-light"

โดยตัว skin ให้กำหนดในส่วนของ components (assetManager) ถ้าอยากได้น้ำตาแบบไหนก็ให้แก้ไข skin ตามที่ชอบได้เลย

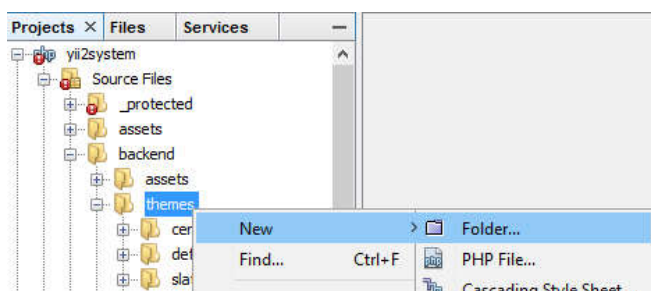
2. ส่วนการกำหนด pathMap และส่วนของ baseUrl ก็กำหนดในส่วนของ components เช่นเดียวกัน (view)

```
'bootstrap' => ['log'],  
'modules' => [],  
'components' => [  
    'assetManager' => [  
        'bundles' => [  
            'dmstr\web\AdminLteAsset' => [  
                'skin' => 'skin-green-light',  
            ],  
        ],  
    ],  
    // here you can set theme used for your backend application  
    // - template comes with: 'default', 'slate', 'spacelab' and 'cerulean'  
    'view' => [  
        'theme' => [  
            'pathMap' => ['@backend/views' => '@backend/themes/adminlte/views'],  
            'baseUrl' => '@web/themes/adminlte',  
        ],  
    ],  
],
```

## การสร้างโฟลเดอร์ Theme สำหรับ AdminLTE

สังเกตเห็นได้ว่าในข้อ 2 Okdหัวข้อที่แล้ว เราได้ตั้งค่าให้กับ pathMap ไว้ ขั้นตอนต่อมาเราก็ต้องสร้าง theme สำหรับใช้กับ adminlte โดยต้องระบุ path ให้ตรงกับที่ตั้งค่าไว้ นั่นคือจะต้องสร้างโฟลเดอร์ชื่อว่า adminlte ในโฟลเดอร์ backend/themes/ นั่นเอง  
ขั้นตอนการสร้าง Theme มีดังนี้

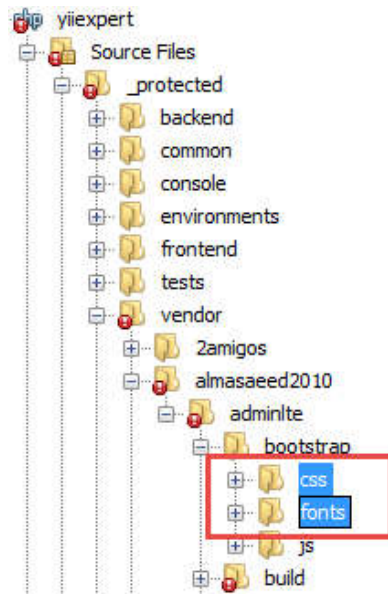
1. สร้างโฟลเดอร์ชื่อว่า adminlte ในโฟลเดอร์ backend/themes/ (คลิกขวาเลือก New->Folder)



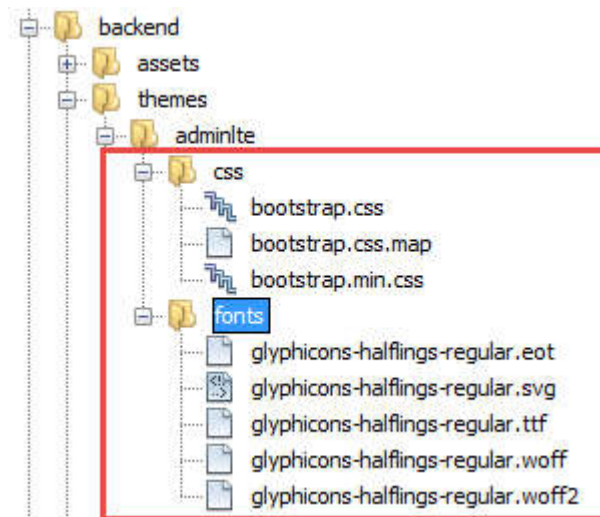
ให้กรอกชื่อ Folder Name ว่า adminlte จากนั้น กด Finish จะได้โฟลเดอร์เปล่าๆ ง่ายๆ



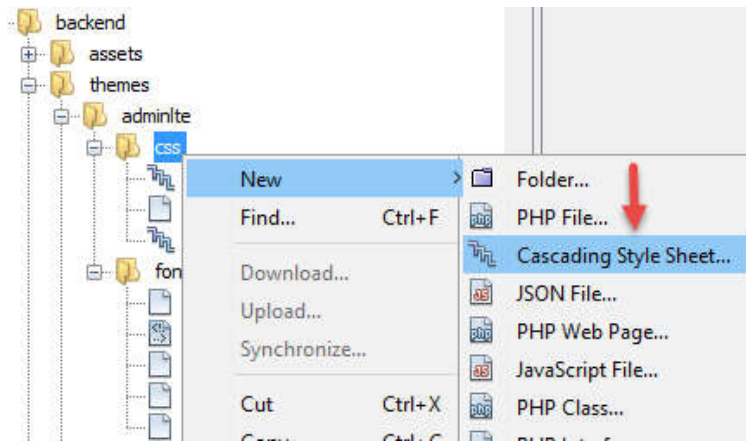
2. ไปที่โฟลเดอร์ `_protected/vendor/almasaeed2010/adminlte/bootstrap/` ให้ Copy โฟลเดอร์ css และ fonts



จากนั้นนำมาวางไว้ที่โฟลเดอร์ `backend/themes/adminlte/` (ที่เพิ่งสร้างไว้ในข้อ 1)

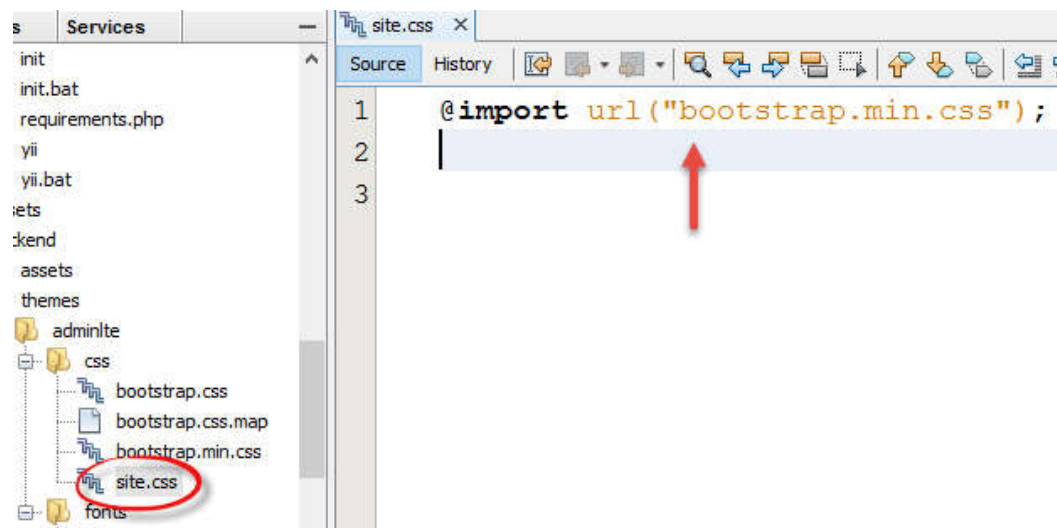


3. สร้างไฟล์ใหม่ชื่อว่า `site.css` ในโฟลเดอร์ `backend/themes/adminlte/css/`



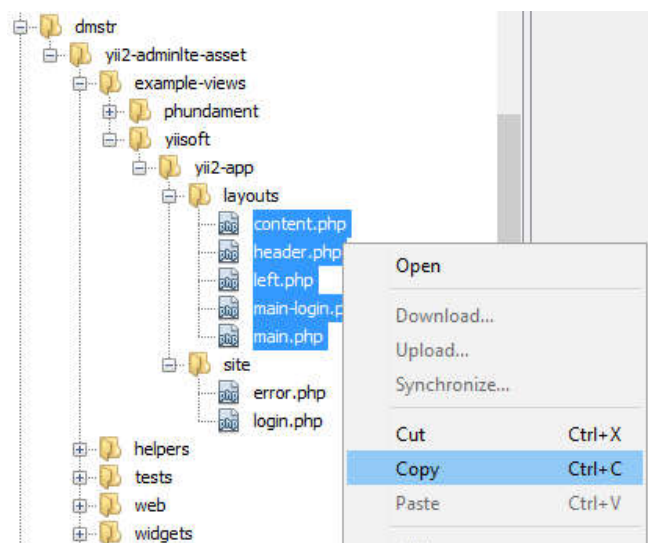
จากนั้นเขียนโค้ดเพื่อ import bootstrap.min.css เข้ามา

@import url("bootstrap.min.css");



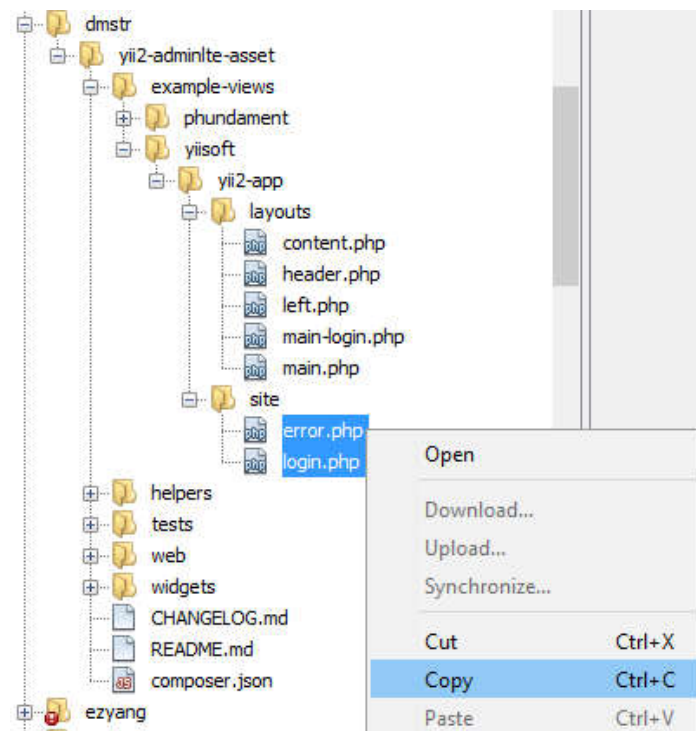
4. เปิดเข้าไปที่โฟลเดอร์ \_protected/vender/dmstr/yii2-adminlte-asset/example-view/yiisoft/yii2-app/ เมื่อเข้าไปแล้วจะเห็นว่า มี 2 โฟลเดอร์ได้แก่ layouts และ site

4.1 Copy ไฟล์ทั้งหมดในโฟลเดอร์ layouts ไปวาง (paste) ที่ backend/views/layouts/ (ให้วางทับไฟล์ main.php ที่มีอยู่เดิมด้วย)

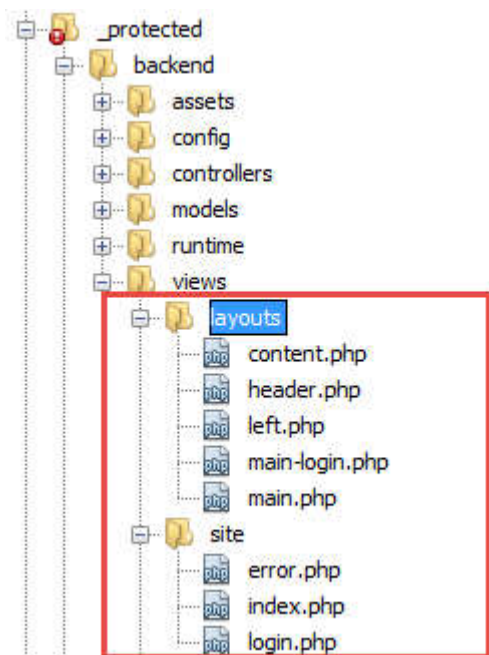




- 4.2 Copy ไฟล์ทั้งหมดในโฟลเดอร์ site ไปวาง (paste) ที่ backend/views/site/ (ให้วางทับไฟล์ login.php และ error.php ที่มีอยู่เดิมด้วย)



- 4.3 เมื่อวางเสร็จแล้วจะได้ตามนี้



5. เปิดไฟล์ backend/views/layouts/main-login.php เพื่อ register App Asset ของ backend โดยแทรกโค้ด

```
AppAsset::register($this);
```

```

main-login.php x
Source History
1 k?php
2 use backend\assets\AppAsset;
3 use yii\helpers\Html;
4
5 /* @var $this \yii\web\View */
6 /* @var $content string */
7
8 AppAsset::register($this);
9
10 dmstr\web\AdminLteAsset::register($this);
11 ?>

```

6. เปิดไฟล์ backend/views/layouts/main.php เพื่อแก้ไขโค้ดในส่วนของ body ให้สามารถเรียกใช้การตั้งค่า skin ได้โดยแก้ไขโค้ดในส่วนของ tag body โดยแก้จาก <body class="hold-transition skin-blue sidebar-mini"> เป็น <body class="<?= \dmstr\helpers\AdminLteHelper::skinClass() ?> sidebar-mini">

```

main.php x
Source History
37 <?php $this->head() ?>
38 </head>
39 <body class="<?= \dmstr\helpers\AdminLteHelper::skinClass() ?> sidebar-mini">
40 <?php $this->beginBody() ?>

```

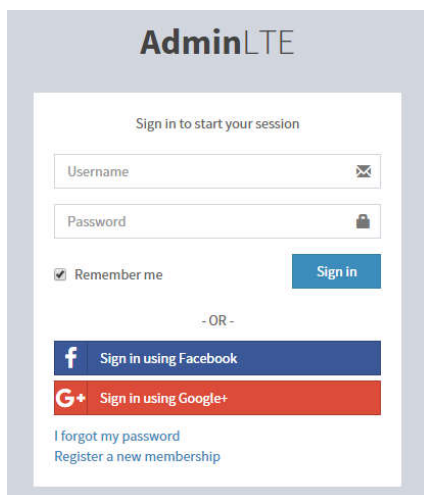
7. บันทึกไฟล์ทั้งหมด แล้วลองรัน <http://localhost/yii2system/backend/> และทดสอบล็อกอินดูครับ เป็นอันเสร็จเรียบร้อย

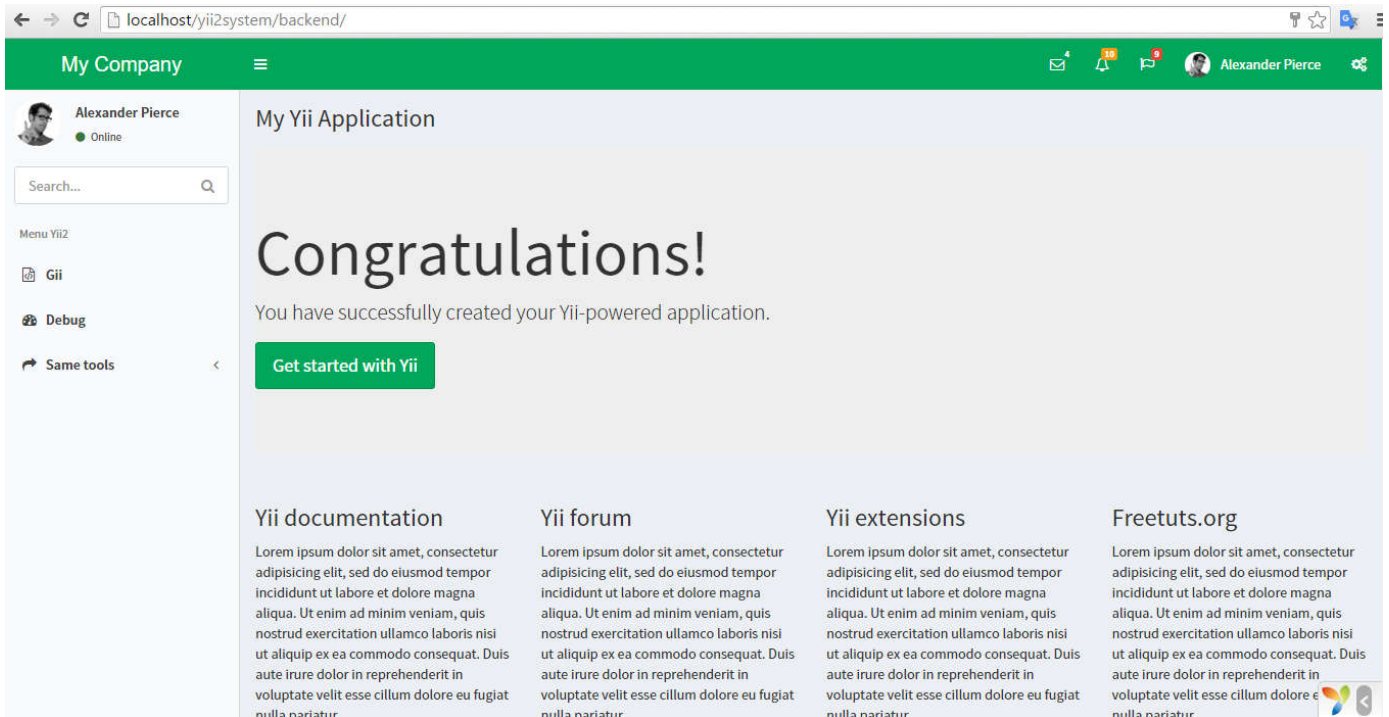
**Note:** ในหน้าล็อกอิน <http://localhost/yii2system/backend/site/login> จะสังเกตเห็นว่าสีหรือไอคอนของ Google+ ไม่มาให้เราเปิดไฟล์ backend/views/site/login.php ขึ้นมาไปที่บรรทัดที่ 60 ให้แก้ไขได้จาก

<a href="#" class="btn btn-block btn-social btn-google-plus btn-flat"><i class="fa fa-google-plus"></i> Sign in using Google+</a>

แก้เป็น

<a href="#" class="btn btn-block btn-social btn-google btn-flat"><i class="fa fa-google-plus"></i> Signin using Google+</a>





หากต้องการปรับแต่งหน้าตาของ AdminLTE แบบเต็มๆ ให้เข้าไปดูรายละเอียดได้ที่

<https://almsaeedstudio.com/themes/AdminLTE/documentation/index.html>

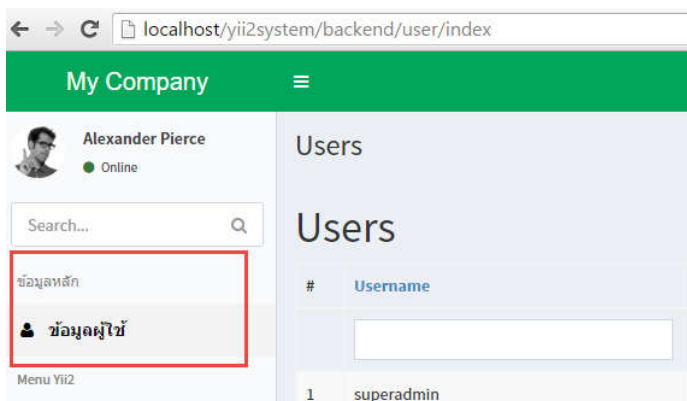
## การเพิ่มเมนูด้านซ้ายให้กับ AdminLTE

การเพิ่มเมนูด้านซ้ายมือ ให้เปิดไฟล์ backend/views/layouts/left.php สามารถปรับแต่ง และแทรกเมนูต่างๆ ได้ที่ dmstr\widgets\Menu::widget โดยทดลองเพิ่ม items ดังนี้

```
['label' => 'ข้อมูลหลัก', 'options' => ['class' => 'header']],
```

```
['label' => 'ข้อมูลผู้ใช้', 'icon' => 'fa fa-user', 'url' => ['/user/index']],
```

บรรทัดแรกกำหนด label และ options โดยระบุ class เป็น header เพื่อเอาไว้จัดกลุ่มเมนูได้ ส่วนบรรทัดที่สอง หากต้องการเพิ่มเมนูต่างๆ ก็สามารถ copy เพิ่มอีกบรรทัดได้เรื่อยๆ โดยสามารถระบุชื่อ icon และ ระบุ url ว่าจะให้ลิงก์ไปที่ Controller อะไรและ action อะไร บันทึกไฟล์แล้วลองรันดูครับ



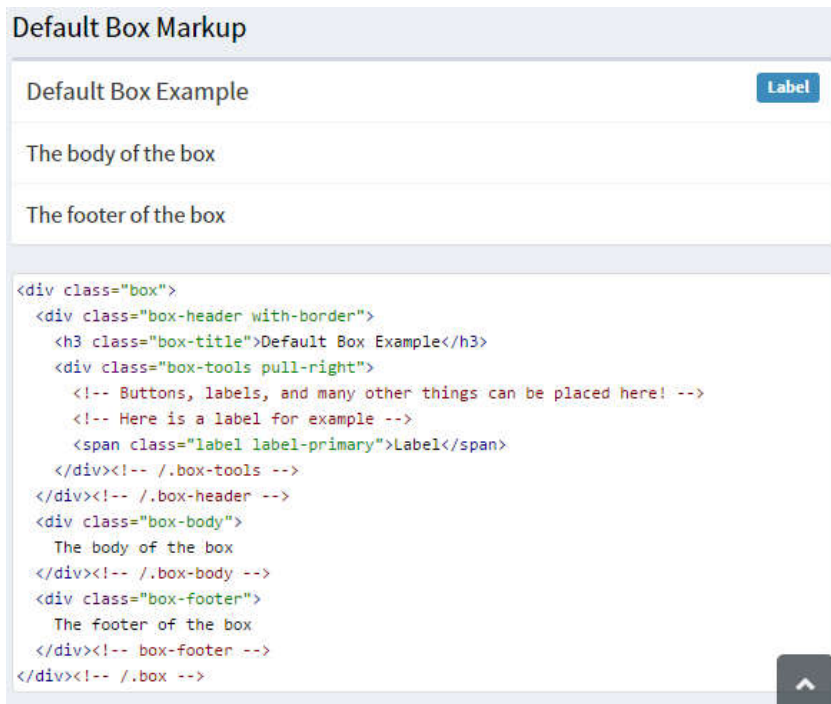
Note: ชื่อไอคอน ใน AdminLTE จะใช้ไอคอนของ Font Awesome ดูรายการไอคอนต่างๆได้ที่

<https://fontawesome.github.io/Font-Awesome/icons/>

## เทคนิคการใส่ Box components ของ AdminLTE ให้กับ GridView และ Form

หากเราดูตามเอกสารของ AdminLTE <https://almsaeedstudio.com/themes/AdminLTE/documentation/index.html#component-box>

ในส่วนของ Box components เราสามารถ Copy และแทรกโค้ด HTML ใส่นั่น Views ได้เลย เพื่อสร้างกล่องสำหรับใส่ข้อความหรืออื่นๆได้ ประกอบด้วยส่วน header, body และ footer นอกจากนี้ยังสามารถกำหนด label (box tools) ได้อีกด้วย



หากเราต้องการแทรก GridView หรือฟอร์ม (Form) ก็สามารถแทรกในส่วน of class ที่ชื่อว่า box-body ได้เลยครับ ยกตัวอย่างเช่น ถ้าผมต้องการแทรก GridView ของ user ก็ให้เปิดไฟล์ backend/views/user/index.php แล้ว Copy ในส่วนของ GridView::widget ทั้งหมดไปวางไว้ใน body ได้เลย หรืออาจปรับแต่งโดยนำปุ่ม Create มาไว้ใน Box ก็ได้ครับ

```
<?php
```

```
use common\helpers\CssHelper;
```

```
use yii\helpers\Html;
```

```
use yii\grid\GridView;
```

```
$this->title = Yii::t('app', 'ข้อมูลผู้ใช้');
```

```
$this->params['breadcrumbs'][] = $this->title;
```

```
?>
```

```

<div class="user-index">

    <div class="box">

        <div class="box-header with-border">

            <h3 class="box-title"><?= Html::a(Yii::t('app', 'เพิ่มผู้ใช้นี้ใหม่'), ['create'], ['class' => 'btn btn-success']) ?></h3>

            <div class="box-tools pull-right">

                <!-- Buttons, labels, and many other things can be placed here! -->

                <!-- Here is a label for example -->

                <span class="label label-primary">Label</span>

            </div><!-- /.box-tools -->

        </div><!-- /.box-header -->

        <div class="box-body">

            <?=

```

```

GridView::widget([

    'dataProvider' => $dataProvider,

    'filterModel' => $searchModel,

    'summary' => false,

    'columns' => [

        ['class' => 'yii\grid\SerialColumn'],

        'username',

        'email:email',

        // status

        [

            'attribute' => 'status',

            'filter' => $searchModel->statusList,

            'value' => function ($data) {

                return $data->statusName;

            },

            'contentOptions' => function($model, $key, $index, $column) {

                return ['class' => CssHelper::statusCss($model->statusName)];

            }

        ],

        // role

```

```

[
    'attribute' => 'item_name',
    'filter' => $searchModel->rolesList,
    'value' => function ($data) {
        return $data->roleName;
    },
    'contentOptions' => function($model, $key, $index, $column) {

        return ['class' => CssHelper::roleCss($model->roleName)];
    }
],
// buttons
['class' => 'yii\grid\ActionColumn',
    'header' => "Menu",
    'template' => '{view} {update} {delete}',
    'buttons' => [
        'view' => function ($url, $model, $key) {
            return Html::a("", $url, ['title' => 'View user',
                'class' => 'glyphicon glyphicon-eye-open']);
        },
        'update' => function ($url, $model, $key) {
            return Html::a("", $url, ['title' => 'Manage user',
                'class' => 'glyphicon glyphicon-user']);
        },
        'delete' => function ($url, $model, $key) {
            return Html::a("", $url, ['title' => 'Delete user',
                'class' => 'glyphicon glyphicon-trash',
                'data' => [
                    'confirm' => Yii::t('app', 'Are you sure you want to delete this user?'),
                    'method' => 'post']
                ]);
        }
    ]
]

```

```

        ]
    ], // ActionColumn

    ], // columns

    ]);

?>

</div><!-- /.box-body -->

<div class="box-footer">

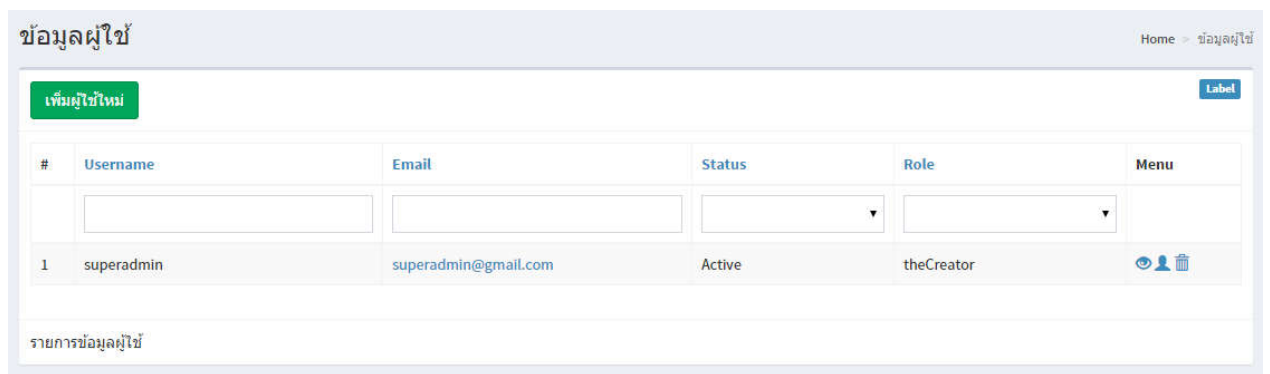
    รายการข้อมูลผู้ใช้

</div><!-- box-footer -->

</div><!-- /.box -->

</div>

```



เมื่อคลิกปุ่ม เพิ่มผู้ใช้ใหม่ ระบบจะเปิดฟอร์มสำหรับเพิ่มผู้ใช้ เราก็สามารถนำ Box components มาใส่เพื่อความสวยงามได้ หรือเมื่อมีการแก้ไขข้อมูลผู้ใช้ ระบบก็จะเรียกฟอร์มแก้ไขขึ้นมาเช่นเดียวกัน วิธีการแทรก box ไม่ว่าจะเป็นการเพิ่ม หรือแก้ไขข้อมูลจะมีการเรียกไฟล์ backend/views/user/\_form.php เข้ามา ประกอบด้วย ถ้าต้องการแทรก box กับการเพิ่มข้อมูลให้แก้ไขไฟล์ backend/views/user/create.php

```

<?php

$this->title = Yii::t('app', 'ข้อมูลผู้ใช้');

$this->params['breadcrumbs'][] = ['label' => Yii::t('app', 'ผู้ใช้งานทั้งหมด'), 'url' => ['index']];

$this->params['breadcrumbs'][] = $this->title;

?>

<div class="user-create">

```

```

<div class="row">

    <div class="col-lg-6">

        <div class="box box-primary">

            <div class="box-header with-border">

                <h3 class="box-title">เพิ่มผู้ใช้นี้ใหม่</h3>

            </div><!-- /.box-header -->

            <div class="box-body">

                <?= $this->render('_form', [

                    'user' => $user,

                    'role' => $role,

                ]) ?>

            </div><!-- /.box-body -->

            <div class="box-footer">

                เป็นการเพิ่มผู้ใช้นี้ในระบบใหม่

            </div><!-- box-footer -->

        </div><!-- /.box -->

    </div>

</div>

</div>

```

ถ้าต้องการแทรก box กับการแก้ไขข้อมูลให้แก้ไขไฟล์ backend/views/user/update.php

```

<?php

$this->title = Yii::t('app', 'แก้ไขผู้ใช้นี้ คุณ ' . ' : ' . $user->username;

$this->params['breadcrumbs'][] = ['label' => Yii::t('app', 'แก้ไข'), 'url' => ['index']];

$this->params['breadcrumbs'][] = ['label' => $user->username, 'url' => ['view', 'id' => $user->id]];

$this->params['breadcrumbs'][] = Yii::t('app', 'แก้ไข');

?>

<div class="user-update">

```



```

<div class="row">

  <div class="col-lg-6">

    <div class="box box-primary">

      <div class="box-header with-border">

        <h3 class="box-title">แก้ไขผู้ใช้</h3>

      </div><!-- /.box-header -->

      <div class="box-body">

        <?= $this->render('_form', [

          'user' => $user,

          'role' => $role,

        ]) ?>

      </div><!-- /.box-body -->

      <div class="box-footer">

        เป็นการแก้ไขผู้ใช้ในระบบ

      </div><!-- box-footer -->

    </div><!-- /.box -->

  </div>

</div>

</div>

```

**Note:** มาถึงตรงนี้เราจะสามารถสร้างเมนูด้านซ้ายได้เอง และสามารถแทรก Box components ได้แล้ว ขอเพิ่มเติมเกี่ยวกับ box นิดหน่อยว่า เราสามารถใส่สีให้กับ box ได้ครับ โดยแทรกโค้ดเข้าไปที่ class box ดังนี้

```

<div class="box box-default">...</div>

<div class="box box-primary">...</div>

<div class="box box-info">...</div>

<div class="box box-warning">...</div>

<div class="box box-success">...</div>

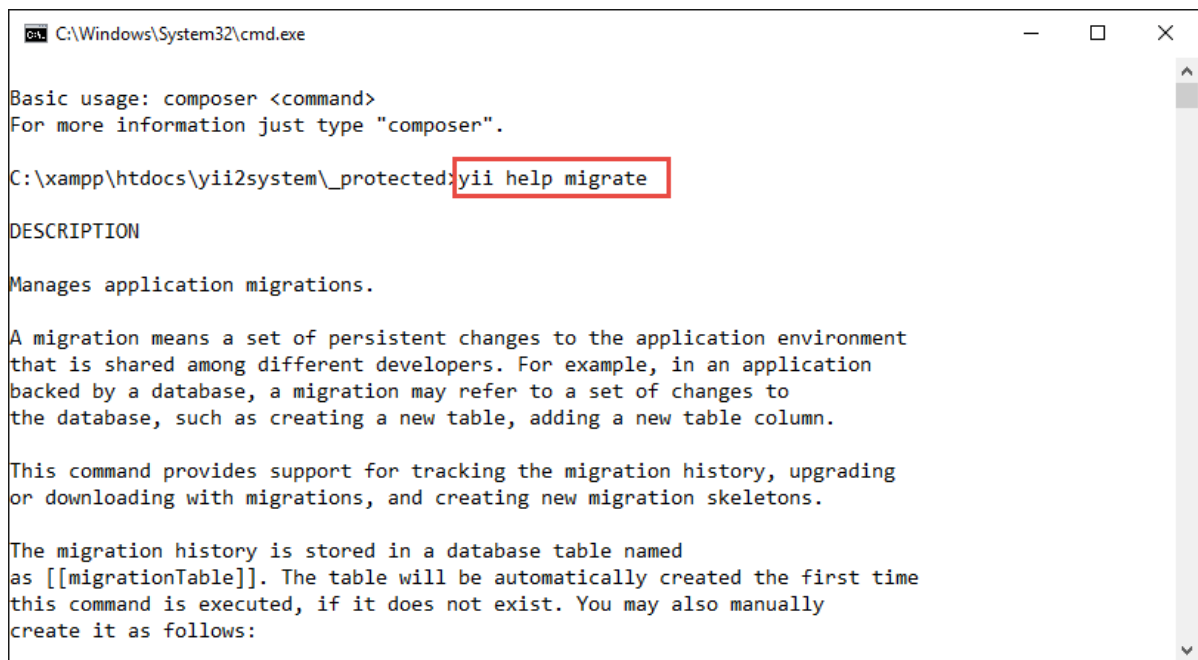
<div class="box box-danger">...</div>

```

## บทที่ 2: Yii2 กับการใช้งานด้านฐานข้อมูล

### การใช้งาน Database Migration

Database Migration เป็นตัวช่วยให้เราสามารถสร้างตารางฐานข้อมูลในระบบ โดยสามารถใช้คำสั่งในการสร้างตารางใหม่ แก้ไข ลบ และ ทำเวอร์ชันของตารางในฐานข้อมูลได้ นอกจากนี้ยังมีคำสั่งให้เราสามารถย้อนกลับไปเวอร์ชันก่อนหน้าได้อีกด้วย การใช้งานให้เข้าไปที่ C:\xampp\htdocs\yii2system\\_protected คลิกขวาแล้วเลือก Use Composer here เหมือนเดิมครับ หากอยากรู้ว่า Database Migration มีคำสั่งอะไรบ้างให้ใช้คำสั่ง `yii help migrate` แล้วกด Enter



```
C:\Windows\System32\cmd.exe

Basic usage: composer <command>
For more information just type "composer".

C:\xampp\htdocs\yii2system\_protected>yii help migrate

DESCRIPTION

Manages application migrations.

A migration means a set of persistent changes to the application environment
that is shared among different developers. For example, in an application
backed by a database, a migration may refer to a set of changes to
the database, such as creating a new table, adding a new table column.

This command provides support for tracking the migration history, upgrading
or downloading with migrations, and creating new migration skeletons.

The migration history is stored in a database table named
as [[migrationTable]]. The table will be automatically created the first time
this command is executed, if it does not exist. You may also manually
create it as follows:
```

#### SUB-COMMANDS

- |                        |                                                          |
|------------------------|----------------------------------------------------------|
| - migrate/create       | Creates a new migration.                                 |
| - migrate/down         | Downgrades the application by reverting old migrations.  |
| - migrate/history      | Displays the migration history.                          |
| - migrate/mark         | Modifies the migration history to the specified version. |
| - migrate/new          | Displays the un-applied new migrations.                  |
| - migrate/redo         | Redoes the last few migrations.                          |
| - migrate/to           | Upgrades or downgrades till the specified version.       |
| - migrate/up (default) | Upgrades the application by applying new migrations.     |

### การสร้างตารางใหม่โดยใช้ Database Migration

การสร้างตารางใหม่ จะใช้คำสั่ง

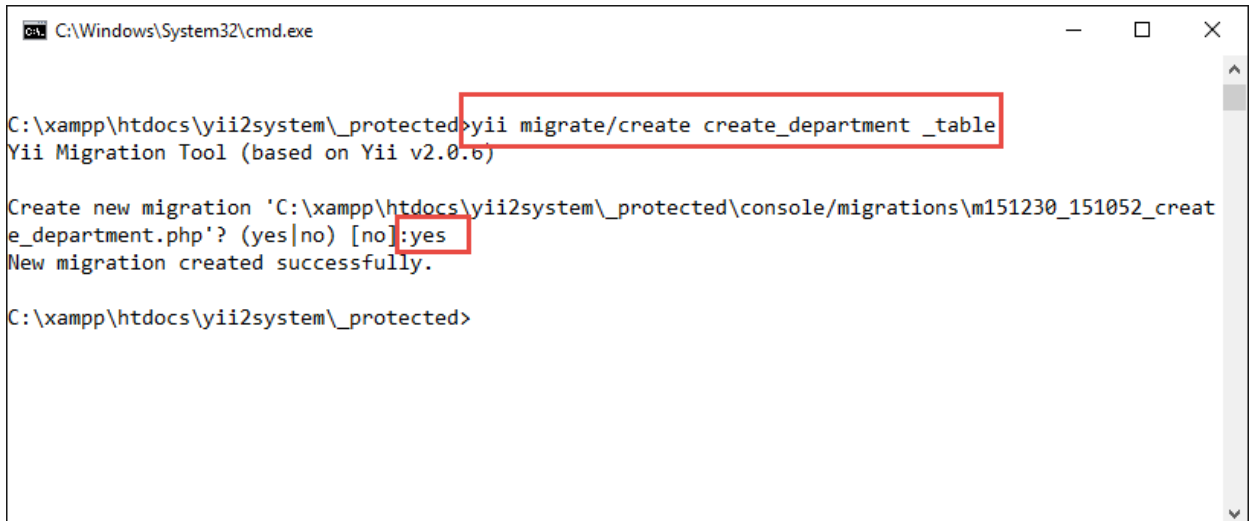
```
yii migrate/create create_ชื่อตาราง_table
```

ยกตัวอย่างเช่น หากเราต้องการสร้างตาราง department ก็ให้ใช้คำสั่ง

```
yii migrate/create create_department_table
```

กด Enter แล้วตอบ yes ระบบจะสร้างไฟล์ .php ให้เรา 1 ไฟล์ที่

C:\xampp\htdocs\yii2system\\_protected\console\Migrations\m151230\_151052\_create\_department.php



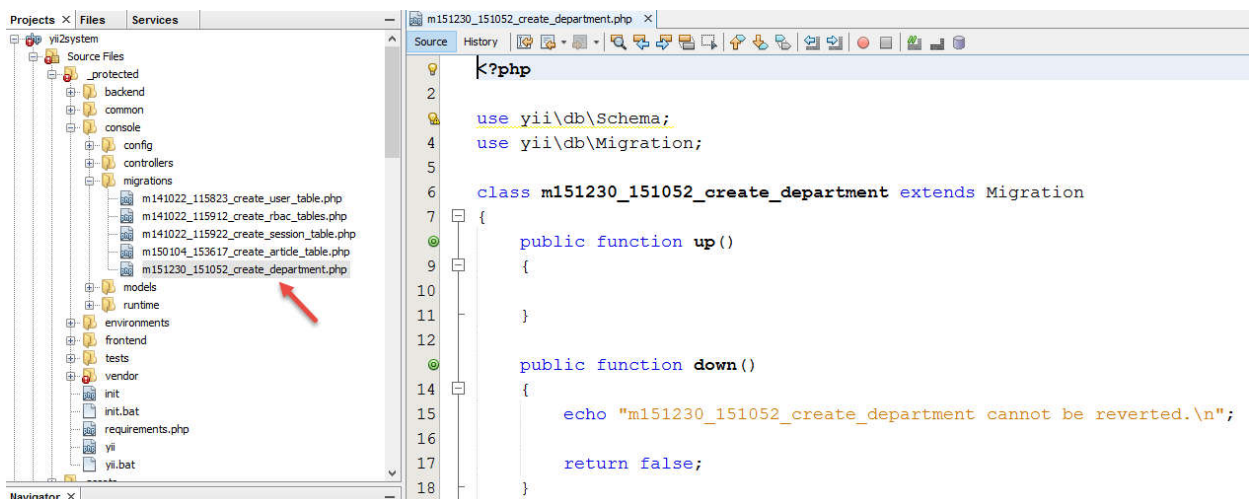
```
C:\Windows\System32\cmd.exe

C:\xampp\htdocs\yii2system\_protected>yii migrate/create create_department_table
Yii Migration Tool (based on Yii v2.0.6)

Create new migration 'C:\xampp\htdocs\yii2system\_protected\console\Migrations\m151230_151052_create_department.php'? (yes|no) [no]:yes
New migration created successfully.

C:\xampp\htdocs\yii2system\_protected>
```

ให้เปิดไฟล์นี้ขึ้นมาเพื่อเริ่มเขียนคำสั่งในการสร้างตารางได้เลย



```
<?php
2
3 use yii\db\Schema;
4 use yii\db\Migration;
5
6 class m151230_151052_create_department extends Migration
7 {
8     public function up()
9     {
10
11     }
12
13     public function down()
14     {
15         echo "m151230_151052_create_department cannot be reverted.\n";
16
17         return false;
18     }
19 }
```

ในคลาสนี้จะประกอบไปด้วย method ที่สำคัญหลักๆ อยู่ 2 method ได้แก่ up และ down ในส่วนของ up จะใช้สำหรับเขียนโค้ดเพื่อสร้างหรือเปลี่ยนแปลงโครงสร้างของตาราง และ down มีไว้สำหรับ revert หรือย้อนกลับนั่นเอง (อาจเป็นคำสั่งสำหรับลบตารางก็ได้) ทดลองเขียนโค้ดทั้งสอง method ดังนี้ และเมื่อสร้างตารางแล้วหากเราต้องการเพิ่มข้อมูลที่หลายๆรายการก็สามารถใช้ batchInsert ได้ครับ

```
public function up()
```

```
{
```

```
    $tableOptions = null;
```

```
    if ($this->db->driverName === 'mysql')
```

```
{
```

```
        $tableOptions = 'CHARACTER SET utf8 COLLATE utf8_unicode_ci ENGINE=InnoDB';
```

```

}

$this->createTable('department', [
    'deptID' => $this->string(2),
    'deptName' => $this->string()->notNull(),
    'PRIMARY KEY (deptID)',
], $tableOptions);

$this->batchInsert('department', ['deptID','deptName'], [
    ['01','การเงิน'],
    ['02','บัญชี'],
    ['03','ไอที'],
]);
}

public function down()
{
    $this->dropTable('department');

    return true;

    // echo "m151105_145451_create_department_table cannot be reverted.\n";

    //return false;
}

```

**Note:** โค้ดในการสร้างตารางนั้นเราจะใช้คำสั่ง createTable เหมือนที่เราคุ้นเคยกันอยู่แล้ว สามารถดูรายละเอียดเพิ่มเติมได้ที่

<http://www.yiiframework.com/doc-2.0/yii-db-schemabuildertrait.html> ตัวอย่างเช่น

```

$this->createTable('example_table', [
    'id' => $this->primaryKey(),
    'name' => $this->string(64)->notNull(),
    'type' => $this->integer()->notNull()->defaultValue(10),
    'description' => $this->text(),
    'rule_name' => $this->string(64),
    'data' => $this->text(),

```

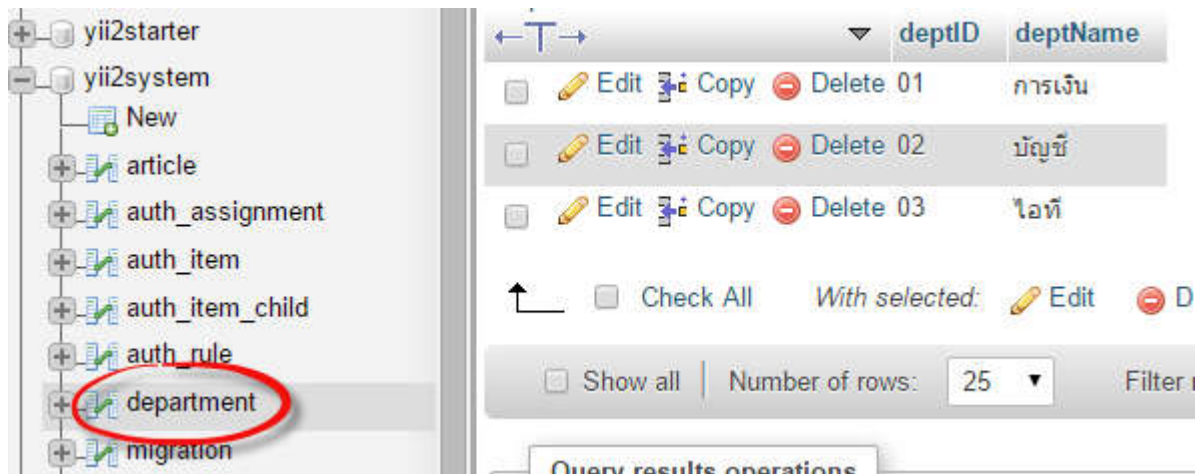
```
'created_at' => $this->datetime()->notNull(),  
'updated_at' => $this->datetime(),  
]);
```

เมื่อเขียนโค้ดเรียบร้อยแล้ว ให้ส่งรันคำสั่งเพื่อสร้างตาราง ดังนี้

yii migrate

```
C:\Windows\System32\cmd.exe  
C:\xampp\htdocs\yii2system\_protected>yii migrate  
Yii Migration Tool (based on Yii v2.0.6)  
  
Total 1 new migration to be applied:  
    m151230_151052_create_department  
  
Apply the above migration? (yes|no) [no]:yes  
*** applying m151230_151052_create_department  
    > create table department ... done (time: 0.441s)  
    > insert into department ... done (time: 0.089s)  
*** applied m151230_151052_create_department (time: 0.833s)  
  
Migrated up successfully.  
C:\xampp\htdocs\yii2system\_protected>
```

ระบบจะทำการสร้างตาราง department และเพิ่มข้อมูลให้เราเรียบร้อยแล้วครับ สามารถเช็คได้จาก phpMyAdmin



## การ Reverting หรือ Undo โดยใช้ Database Migration

หลังจากที่สร้างตารางใหม่ไปเรียบร้อยแล้ว บางครั้งเราอยากแก้ไขโครงสร้างของตารางใหม่ หรือเพิ่มข้อมูลใหม่ ก็สามารถที่จะ revert หรือ undo ได้ โดยสามารถเขียนได้ที่ method down นั่นเอง โดยการ revert นั้นเราได้เขียนโค้ดกันไปแล้ว โดยจะให้สั่ง drop หรือ ลบตารางออก นั่นเอง

คำสั่งในการ revert นั้น สามารถเขียนได้ ดังนี้

yii migrate/down

```
C:\Windows\System32\cmd.exe
C:\xampp\htdocs\yii2system\_protected>yii migrate/down
Yii Migration Tool (based on Yii v2.0.6)

Total 1 migration to be reverted:
    m151230_151052_create_department

Revert the above migration? (yes|no) [no]:yes
*** reverting m151230_151052_create_department
    > drop table department ... done (time: 0.221s)
*** reverted m151230_151052_create_department (time: 0.307s)

Migrated down successfully.

C:\xampp\htdocs\yii2system\_protected>
```

ทดสอบโดยการเปิดดูตารางใน phpMyAdmin ถ้าการทำงานเรียบร้อยก็สังเกตว่าตาราง department ได้ลบไปเรียบร้อยแล้ว

**Note:** ในการสร้าง Web Application เราควรวางแผนและออกแบบฐานข้อมูลให้เรียบร้อยก่อน สามารถใช้ Database Migration นี้ช่วยได้ครับ ดูรายละเอียดเพิ่มเติมได้ที่ <http://www.yiiframework.com/doc-2.0/guide-db-migrations.html>

## การใช้งาน Data Access Object

Data Access Object หรือ DAO เป็นคลาสที่เอาไว้จัดการเรื่องฐานข้อมูลให้ใช้งานง่าย และสะดวกขึ้นกว่าการเขียนแบบ PDO ปกติ โดยสามารถใส่คำสั่ง SQL และได้ผลลัพธ์ในรูปแบบของ PHP Arrays ได้เลย นอกจากนี้ยังสามารถรองรับฐานข้อมูลได้หลายตัว ดังนี้

MySQL          MariaDB          SQLite          PostgreSQL          CUBRID: version 9.3 or higher.

Oracle          MSSQL: version 2008 or higher.

### Data Access Object: สร้างการติดต่อกับฐานข้อมูล

การติดต่อกับฐานข้อมูลใน Yii นั้นเราสามารถเข้าไปกำหนดค่าได้ที่ไฟล์ common/config/main-local.php (เรากำหนดเรียบร้อยแล้ว) แต่หากต้องการติดต่อกับฐานข้อมูลยี่ห้ออื่นๆ เช่น Oracle, MS SQL Server เป็นต้น ก็สามารถทำได้เช่นเดียวกัน โดยให้กำหนดที่ dsn นั้นเอง

```
'dsn' => 'mysql:host=localhost;dbname=yii2system',
```

- MySQL, MariaDB: `mysql:host=localhost;dbname=mydatabase`
- SQLite: `sqlite:/path/to/database/file`
- PostgreSQL: `pgsql:host=localhost;port=5432;dbname=mydatabase`
- CUBRID: `cubrid:dbname=demodb;host=localhost;port=33000`
- MS SQL Server (via sqlsrv driver): `sqlsrv:Server=localhost;Database=mydatabase`
- MS SQL Server (via dblib driver): `dblib:host=localhost;dbname=mydatabase`
- MS SQL Server (via mssql driver): `mssql:host=localhost;dbname=mydatabase`
- Oracle: `oci:dbname=//localhost:1521/mydatabase`

ถ้าหากเราต้องการติดต่อกับฐานข้อมูลนอกเหนือจากนี้เช่น Access หรือ DBF ก็สามารถทำผ่าน ODBC ได้ครับ

ในบางครั้งเราต้องการรันคำสั่ง SQL (createCommand) บางคำสั่งหลังจากการติดต่อฐานข้อมูลกับฐานข้อมูลแล้ว ก็สามารถทำได้โดยเพิ่ม event ที่ชื่อว่า afterOpen (เพิ่มในส่วนของภายใน db components) ดังนี้

```
'db' => [
    // ...

    'on afterOpen' => function($event) {
        // $event->sender refers to the DB connection

        $event->sender->createCommand("SET time_zone = 'UTC'")->execute();
    }
],
```

**Note:** ดูรายละเอียดการเชื่อมต่อกับฐานข้อมูลทั้งหมดได้ที่

<http://www.yiiframework.com/doc-2.0/guide-db-dao.html#creating-db-connections>

## Data Access Object: การส่งรันคำสั่ง SQL

การส่งรัน SQL นั้น DAO จะใช้อ็อบเจกต์ชื่อว่า `yii\db\Command` โดยสร้างผ่าน static method `yii\db\Connection::createCommand()` อีกทีครับ เมื่อเราส่งรัน SQL เสร็จแล้ว ก็ต้องเรียกดูข้อมูลขึ้นมา หรือการ query ข้อมูลนั่นเอง โดยการใช้ method ต่างๆ ยกตัวอย่าง เช่น `queryAll()`, `queryOne()` เป็นต้น

## Data Access Object: เมธอดสำหรับ query ข้อมูล

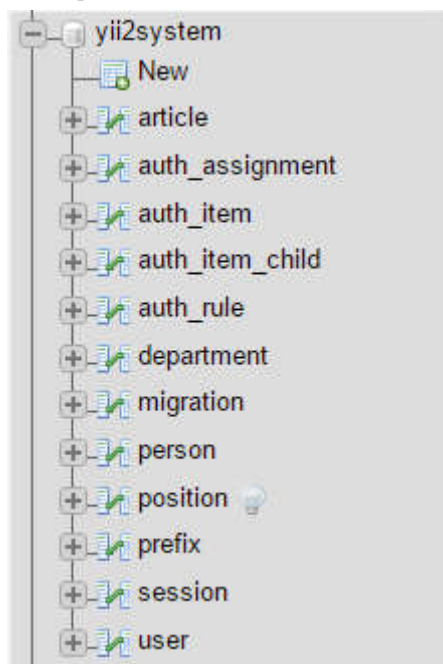
รายละเอียด Method ที่ใช้ดึง หรือเรียกดูข้อมูลจากรันคำสั่ง SQL แล้ว มีดังนี้

- queryAll() ผลลัพธ์จะได้ record ทั้งหมดของการคิวรี ถ้าคิวรีแล้วไม่พบข้อมูล จะได้ค่า array ที่ว่างเปล่า (empty)
- queryOne() จะคืนค่าผลลัพธ์แถวที่ 1 เท่านั้น ถ้าคิวรีแล้วไม่พบข้อมูลจะคืนค่าเป็น false (Boolean)
- queryScalar() จะคืนค่าผลลัพธ์เป็นคอลัมน์ที่ 1 และ แถวที่ 1 เท่านั้น (cell เดียว) ถ้าคิวรีแล้วไม่พบข้อมูลจะคืนค่าเป็น false (Boolean)
- query() เป็นการส่งคิวรีได้ทั่วไป โดยจะคืนค่าเป็น yii\db\DataReader object

ตั้งแต่หัวข้อต่อไปนี้ ผมได้อัปเดตไฟล์ yii2system.sql ซึ่งจะเป็นตัวอย่างตาราง และข้อมูลสำหรับการเรียนรู้ไว้ให้แล้ว โดยให้ไปดาวน์โหลดได้ที่ [www.codingthailand.com/yii2system.zip](http://www.codingthailand.com/yii2system.zip)

จากนั้น ให้ทุกคน Import ไฟล์ตาราง เข้ามาฐานข้อมูลของเราได้เลย โดยมีขั้นตอน ดังนี้

1. เปิด phpMyAdmin แล้วคลิกที่ฐานข้อมูล yii2system
2. คลิกที่เมนู Import -> เลือก Choose File -> เสร็จแล้วกดปุ่ม Go
3. รอสักครู่ เราจะได้ตารางใหม่ทั้งหมด 4 ตาราง ได้แก่ department, prefix, position, person



**Data Access Object:** ลองเขียน DAO เพื่อเรียกดู และแสดงข้อมูลจากตาราง position

1. สร้างไฟล์ใหม่ชื่อ PositionController.php ที่ backend/controllers/ เขียนโค้ด ดังนี้

```
<?php
```



```
namespace backend\controllers;
```

```
use yii\web\Controller;
```

```
use yii\filters\AccessControl;
```

```
use yii\filters\VerbFilter;
```

```
use Yii;
```

```
/**
```

```
 * Site controller.
```

```
 * It is responsible for displaying static pages, and logging users in and out.
```

```
 */
```

```
class PositionController extends Controller {
```

```
/**
```

```
 * Returns a list of behaviors that this component should behave as.
```

```
 *
```

```
 * @return array
```

```
 */
```

```
public function behaviors() {
```

```
    return [
```

```
        'access' => [
```

```
            'class' => AccessControl::className(),
```

```
            'rules' => [
```

```
                [
```

```
                    'actions' => ['login', 'error'],
```

```
                    'allow' => true,
```

```
                ],
```

```
                [
```

```
                    'actions' => ['logout', 'index'],
```

```
                    'allow' => true,
```

```
                    'roles' => ['@'],
```

```
                ],
```

```

        ],
    ],
    'verbs' => [
        'class' => VerbFilter::className(),
        'actions' => [
            'logout' => ['post'],
        ],
    ],
];
}

```

```

/**
 * Declares external actions for the controller.
 *
 * @return array
 */

```

```

public function actions() {
    return [
        'error' => [
            'class' => 'yii\web\ErrorAction',
        ],
    ];
}

```

```

public function actionIndex() {

```

```

    $position = Yii::$app->db->createCommand("SELECT * FROM position ORDER BY position_name ASC")->queryAll();

```

```

    $count = Yii::$app->db->createCommand("SELECT COUNT(*) FROM position")->queryScalar();

```

```

    return $this->render('index', [
        'positions' => $position,
    ],

```

```

        'count' => $count,

    ));

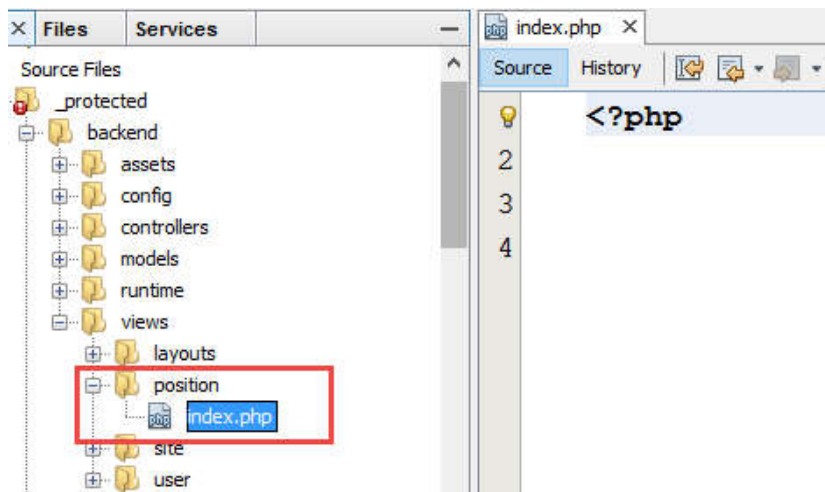
}

}

```

ในส่วนของเมธอด actionIndex การเรียกใช้ DAO นั้นง่ายมากเพียงแค่เราใส่คำสั่ง SQL เข้าไปใน method createCommand แล้วคิวรีข้อมูลตามประเภทของการคิวรี เช่น queryAll() หรือ queryScalar() สำหรับการ count เป็นต้น เมื่อคิวรีเรียบร้อยแล้ว เราส่งค่าตัวแปรไปที่ backend/views/position/index.php สองตัวได้แก่ \$position ที่เก็บข้อมูลจากการ SELECT และ \$count ซึ่งเก็บข้อมูลจากการ count หรือนับจำนวนแถวทั้งหมด

2. ต่อมาในส่วนของ views ให้สร้างไฟล์เดือริใหม่ชื่อว่า position เสร็จแล้วสร้างไฟล์ index.php ขึ้นมาเพื่อรับค่าจาก PositionController



เขียนโค้ดในไฟล์ index.php เพื่อแสดงผลข้อมูล ดังนี้

```

<?php

use yii\helpers\Html;

use yii\helpers\Url;

$this->title = Yii::t('app', 'ตำแหน่ง (ทั้งหมด )'.Html::encode($count). ' รายการ');

$this->params['breadcrumbs'][] = $this->title;

?>

<div class="row">

    <div class="col-md-6">

        <table class="table table-hover">

```

```

<tr>
    <th>รหัส</th>
    <th>ชื่อตำแหน่ง</th>
    <th>ลบ</th>
</tr>

<?php foreach ($positions as $position): ?>
<tr>
    <td><? = Html::encode($position['position_id']); ?></td>
    <td><? = Html::encode($position['position_name']); ?></td>
    <td><a href="<? = Url::to(['position/delete', 'id'=>Html::encode($position['position_id']) ]; ?>"><span class="fa fa-
trash"></span></a></td>
</tr>
<?php endforeach; ?>
</table>
</div>
</div>

```

บันทึกไฟล์ แล้วลองรันดู <http://localhost/yii2system/backend/position/index>

รหัส	ชื่อตำแหน่ง	ลบ
101	ช่างทอ	
22	ช่างเชื่อม	
20	ช่างไฟฟ้า	
17	ช่างไม้	
99	ช่างไม้ครุฑเทพ	

อธิบายได้เพิ่มเติม เราใช้การวนลูปตัวแปร \$positions ซึ่งเป็น array โดยใช้คำสั่ง foreach และมีการเพิ่มไอคอนเพื่อจะทำลิงก์สำหรับลบข้อมูลอีกด้วย โดยเราจะส่งรหัสตำแหน่ง เพื่อไปลบข้อมูลนั่นเอง

**Note:** อย่าลืมว่าเราสามารถสร้างเมนูด้านซ้ายมือได้โดยเปิดไฟล์ backend/views/layouts/left.php เพื่อเพิ่มเมนู Position หรือ ตำแหน่งงานได้ เช่น ['label' => 'ข้อมูลตำแหน่ง', 'icon' => 'fa fa-dashboard', 'url' => ['/position/index']],

**Data Access Object:** ลองเขียน DAO เพื่อลบจากตาราง position

จากหัวข้อที่แล้ว เราได้ลิงก์และไอคอนสำหรับลบข้อมูลโดยส่งรหัสตำแหน่ง หรือ id ไปด้วย

```
<a href="<?= Url::to(['position/delete','id'=>Html::encode($position['position_id']) ]; ?>"><span class="fa fa-trash"></span></a>
```

ขั้นตอนการลบข้อมูลให้เปิดไฟล์ PositionController.php เพื่อเพิ่ม method สำหรับการลบข้อมูล ดังนี้ (อาจเพิ่มหลัง actionIndex ก็ได้)

```
public function actionDelete($id) {  
    Yii::$app->db->createCommand()->delete('position', ['position_id' => $id])->execute();  
    return $this->redirect(['index']);  
}
```

จากนั้นให้เพิ่มโค้ด เพื่อกำหนดสิทธิ์ให้สามารถเรียกใช้ action delete ได้ ใน method behaviors() ดังนี้ ส่วนเรื่องการกำหนดสิทธิ์จะอธิบายในภายหลัง

```
public function behaviors() {  
    return [  
        'access' => [  
            'class' => AccessControl::className(),  
            'rules' => [  
                [  
                    'actions' => ['login', 'error'],  
                    'allow' => true,  
                ],  
                [  
                    'actions' => ['logout', 'index', 'delete'],  
                    'allow' => true,  
                    'roles' => ['@'],  
                ],  
            ],  
        ],  
    ],  
];
```

```

],
'verbs' => [
    'class' => VerbFilter::className(),
    'actions' => [
        'logout' => ['post'],
    ],
],
];
}

```

เสร็จแล้วบันทึกไฟล์ แล้วทดสอบการลบข้อมูลดูนะครับ การทำงานของ method นี้คือเราส่ง \$id เข้ามาทำงานแล้วเรียกใช้ method delete จากนั้นเรียก execute() เพื่อลบข้อมูลตามรหัสตำแหน่ง โค้ดทั้งหมดของ PositionController.php จะได้เป็น ดังนี้

```

<?php
namespace backend\controllers;

use yii\web\Controller;
use yii\filters\AccessControl;
use yii\filters\VerbFilter;
use Yii;

/**
 * Site controller.
 * It is responsible for displaying static pages, and logging users in and out.
 */
class PositionController extends Controller {

    /**
     * Returns a list of behaviors that this component should behave as.
     *
     * @return array
     */
}

```

```

public function behaviors() {
    return [
        'access' => [
            'class' => AccessControl::className(),
            'rules' => [
                [
                    'actions' => ['login', 'error'],
                    'allow' => true,
                ],
                [
                    'actions' => ['logout', 'index', 'delete'],
                    'allow' => true,
                    'roles' => ['@'],
                ],
            ],
        ],
        'verbs' => [
            'class' => VerbFilter::className(),
            'actions' => [
                'logout' => ['post'],
            ],
        ],
    ];
}

```

```
/**
```

```
 * Declares external actions for the controller.
```

```
 *
```

```
 * @return array
```

```
 */
```

```
public function actions() {
```

```
    return [
```

```

        'error' => [
            'class' => 'yii\web\ErrorAction',
        ],
    ];
}

```

```

public function actionIndex() {

```

```

    $position = Yii::$app->db->createCommand("SELECT * FROM position ORDER BY position_name ASC")->queryAll();

```

```

    $count = Yii::$app->db->createCommand("SELECT COUNT(*) FROM position")->queryScalar();

```

```

    return $this->render('index', [
        'positions' => $position,
        'count' => $count,
    ]);
}

```

```

public function actionDelete($id) {

```

```

    Yii::$app->db->createCommand()->delete('position', ['position_id' => $id])->execute();

```

```

    return $this->redirect(['index']);

```

```

}

```

```

}

```

## Data Access Object: DAO สำหรับการเพิ่ม และแก้ไขข้อมูล

ส่วนใหญ่แล้วปกติเราจะไม่ค่อยได้ใช้ DAO ในการเพิ่ม หรือแก้ไขข้อมูลครับ เพราะยุ่งยากในการเขียน และการสร้างฟอร์มต่างๆ ซึ่งเสียเวลา  
 มาก ปกติเราจะใช้ Active Record ในการทำงานส่วนนี้ สำหรับการเพิ่ม หรือแก้ไข เราจะใช้ Gii ในการสร้างโค้ดให้ครับ ซึ่งจะขอพูดถึงใน  
 หัวข้อต่อไป ถ้าเราสังเกตการลบข้อมูลจากหัวข้อที่แล้ว จะเห็นว่า เวลารันคำสั่งจะใช้ execute() และคำสั่ง insert, update ก็ใช้ execute()  
 เช่นเดียวกัน ตัวอย่างเช่น

```

$connection->createCommand()->insert('user', [ 'name' => 'Sam', 'age' => 30, ])->execute(); หรือ

```

```

$connection->createCommand()->update('user', ['status' => 1], 'age > 30')-> execute();

```



สรุปก็คือ เราจะใช้ execute() สำหรับคำสั่ง insert, update, delete และ query ทั้งหลายใช้กับการ select หรือเรียกดูข้อมูลนั่นเอง

**Data Access Object:** การใช้เทคนิค prepare เพื่อทำให้การเรียกดูข้อมูลมีประสิทธิภาพมากขึ้น

บางครั้งเวลาเราใช้ DAO เพื่อคิวรีข้อมูลเยอะๆ อาจทำให้ประสิทธิภาพการเรียกดูข้อมูลลดลงได้ และเพื่อให้ประสิทธิภาพดีขึ้น แนะนำให้ใส่คำสั่ง prepare โดยกำหนด argument เป็น true ไว้ก่อนเรียกใช้ createCommand ครับ ดังนี้

```
public function actionIndex() {  
    Yii::$app->db->createCommand()->prepare(true);  
    $position = Yii::$app->db->createCommand("SELECT * FROM position ORDER BY position_name ASC")->queryAll();  
    $count = Yii::$app->db->createCommand("SELECT COUNT(*) FROM position")->queryScalar();  
    return $this->render('index', [  
        'positions' => $position,  
        'count' => $count,  
    ]);  
}
```

**Data Access Object:** การป้องกันไม่ให้ชื่อคอลัมน์ในตาราง และชื่อตารางซ้ำกับคำสั่งของฐานข้อมูล

เพื่อป้องกันการปวดหัวกับชื่อคอลัมน์หรือชื่อตารางที่บางครั้งเราอาจเผลอไปตั้งซ้ำกับคำสั่งของฐานข้อมูล และทำให้เกิด errors แบบคาดไม่ถึงได้ ก็ควรใส่เครื่องหมายดังนี้

- สำหรับชื่อคอลัมน์ ให้ครอบด้วยก้ามปู (double square brackets) เช่น [[column name]]
- สำหรับชื่อตาราง ให้ครอบด้วยปีกกา (double curly brackets) เช่น {{table name}}

แต่ถ้ามั่นใจอยู่แล้วก็ไม่จำเป็นต้องใส่ก็ได้ครับ ตัวอย่าง

```
Yii::$app->db->createCommand("SELECT COUNT([[id]]) FROM {{position}}")->queryScalar();
```

**สรุป** การใช้ DAO ข้อดี คือเราสามารถ copy SQL จากที่อื่นๆ วางและสั่งรันได้ทันที เหมาะกับ SQL ที่ซับซ้อน เช่น การทำรายงาน เป็นต้น แต่ข้อเสียคือ บางครั้ง SQL นั้นอาจไม่ได้ใช้กับฐานข้อมูลได้ทุกยี่ห้อ ทางแก้ คือ เราจะมาเรียนรู้การใช้ Query Builder ซึ่งจะตอบโจทย์ตรงนี้ด้วยครับ

## ทำความรู้จักกับ Query Builder

Query Builder เป็นคลาสที่ใช้ในการจัดการกับฐานข้อมูลต่างๆ และเป็นการเขียนคลาสครอบตัว DAO อีกที เพื่อช่วยให้เราสามารถเขียนคำสั่ง SQL ในรูปแบบเหมือนกับการเขียนโปรแกรม และที่สำคัญคำสั่งต่างๆ นั้นรองรับทุกฐานข้อมูล!! ประโยชน์ของมันคือ จะทำให้โค้ดของเราอ่านง่ายขึ้น เพราะอยู่ในรูปแบบ Programming และยังช่วยให้โค้ดที่เราเขียนขึ้นมามีความปลอดภัยและสะดวกมากขึ้น เพราะไม่ต้องมา bind parameter เพื่อป้องกัน SQL Injection เองเหมือน DAO ครับ

## ขั้นตอนการใช้ Query Builder

ขั้นตอนการใช้งาน Query Builder นั้นมีเพียงแค่ 2 ขั้นตอนเท่านั้น ได้แก่

1. สร้าง object yii\db\Query เพื่อเรียกใช้คำสั่ง SQL เช่น select, where, order by เป็นต้น โดยลำดับการเขียนจะเรียงเหมือนเราเขียน SQL แบบปกติ (อย่าลืม use yii\db\Query; เข้ามาในไฟล์ก่อน)
2. ขั้นสุดท้ายอย่าลืมคิวรีข้อมูลหลังจากเรียก method เช่น all(), scalar() เป็นต้น เพื่อดึงข้อมูลจากฐานข้อมูล (เหมือน DAO)

ตัวอย่างการเขียน Query Builder

```
$query = new Query();  
$query->select(['id', 'email'])  
->from('user')  
->where(['last_name' => 'Smith'])  
->limit(10)  
->all();
```

1

2

**Note:** การเขียน Query Builder นั้นง่ายมาก ให้เราเรียงลำดับการเขียนเหมือนเขียน SQL ปกติเลย เช่น select id,email from user where 'last\_name' = 'Smith' LIMIT 10; ก็จะได้ Query Builder แบบรูปด้านบนเลยครับ

## ฝึกเขียน Query Builder และ ดูรายละเอียด Method สำหรับการคิวรีข้อมูล

ในส่วนของแบบฝึกหัด Query Builder เราจะใช้ตาราง Department ในการใช้งานครับ มีขั้นตอน ดังนี้

1. ให้สร้างไฟล์ DepartmentController.php ขึ้นมาที่โฟลเดอร์ backend/controllers/ แล้วพิมพ์โค้ด ดังนี้

```
<?php
```

```
namespace backend\controllers;
```

```
use yii\web\Controller;
```

```
use yii\filters\AccessControl;
```

```
use yii\filters\VerbFilter;
```

```
use Yii;
```

```
use yii\db\Query;
```

```
class DepartmentController extends Controller {
```

```
    public function behaviors() {
```

```
        return [
```

```
            'access' => [
```

```
                'class' => AccessControl::className(),
```

```
                'rules' => [
```

```
                    [
```

```
                        'actions' => ['login', 'error'],
```

```
                        'allow' => true,
```

```
                    ],
```

```
                    [
```

```
                        'actions' => ['logout', 'index'],
```

```
                        'allow' => true,
```

```
                        'roles' => ['@'],
```

```
                    ],
```

```
                ],
```

```
        ],
```

```
        'verbs' => [
```

```
            'class' => VerbFilter::className(),
```

```
            'actions' => [
```

```
                'logout' => ['post'],
```

```

        ],
    ],
];
}

public function actions() {

    return [

        'error' => [

            'class' => 'yii\web\ErrorAction',

        ],

    ];

}

public function actionIndex() {

    $query = new Query();

    //จะพิมพ์โค้ดตรงนี้เพื่อทำแบบฝึกหัด method ต่างของ Query Builder โดยจะมีการส่ง departments และ count ไปที่ views

    return $this->render('index', [

        'departments' => $department,

        'count' => $count,

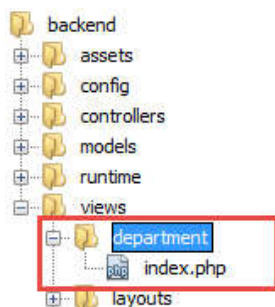
    ]);

}

}

```

2.สร้างโฟลเดอร์ department และไฟล์ index.php สำหรับส่วนแสดงผล (views)



เปิดไฟล์ index.php แล้วพิมพ์โค้ด เพื่อแสดงผลข้อมูลที่เกิดจากการ query ดังนี้

```
<?php
```

```
use yii\helpers\Html;
```

```
use yii\helpers\Url;
```

```
$this->title = Yii::t('app', 'แผนก ทั้หมด ' . Html::encode($count) . ' รายการ');
```

```
$this->params['breadcrumbs'][] = $this->title;
```

```
?>
```

```
<div class="row">
```

```
    <div class="col-md-6">
```

```
        <table class="table table-hover">
```

```
            <tr>
```

```
                <th>รหัส</th>
```

```
                <th>แผนก</th>
```

```
                <th>ลบ</th>
```

```
            </tr>
```

```
            <?php foreach ($departments as $department): ?>
```

```
                <tr>
```

```
                    <td><?= Html::encode($department['department_id']); ?></td>
```

```
                    <td><?= Html::encode($department['department_name']); ?></td>
```

```
                    <td><a href="<?= Url::to(['department/delete', 'id'=>Html::encode($department['department_id'])]); ?>"><span  
class="fa fa-trash"></span></a></td>
```

```
                </tr>
```

```
            <?php endforeach; ?>
```

```
        </table>
```

```
    </div>
```

```
</div>
```

ต่อไปถ้าเขียนคำสั่งแบบฝึกหัด ใน actionIndex() กัน โดยสามารถ comment คำสั่งเก่าๆ ที่เคยเขียนไว้เก็บไว้ดูได้ครับ มาเริ่มกันเลย!

- เมธอด count() ใช้เมื่อเราต้องการนับจำนวนแถวทั้งหมด หรือเงื่อนไขมาจากการคิวรี่ ตัวอย่าง

```
$count = $query->from('department')->count();
```

- เมธอด select() ใช้เมื่อคิวรี่ข้อมูลทั้งหมด หรือระบुकอลัมน์ของตารางได้ โดยการระบุคอลลัมน์ตาราง สามารถกำหนดเป็นรูปแบบได้ทั้งแบบ array หรือ string ก็ได้ ตัวอย่าง

`$department = $query->select()->from('department')->all();` //select ข้อมูลทุกคอลัมน์ และทั้งหมด

ถ้าอยากระบุคอลัมน์ที่อยากได้ ก็สามารถระบุชื่อคอลัมน์เข้าไปข้างใน `select()` ได้เลยจะเป็น array หรือ string ก็ได้ครับ

ระบุคอลัมน์ แบบ array

```
$department = $query->select(['department_id','department_name'])
    ->from('department')
    ->all();
```

ระบุคอลัมน์แบบ string

```
$department = $query->select('department_id,department_name')
    ->from('department')
    ->all();
```

ลองพิมพ์โค้ด ใน `actionIndex()` ดังนี้

```
public function actionIndex() {

    $query = new Query();

    $count = $query->from('department')->count();

    /*$department = $query->select()
        ->from('department')
        ->all();*/

    $department = $query->select(['department_id','department_name'])
        ->from('department')
        ->all();

    return $this->render('index', [
        'departments' => $department,
        'count' => $count,
    ]);
}
```

บันทึกไฟล์ เสร็จแล้วทดลองรันดูครับ <http://localhost/yii2system/backend/department/index>

← → ↻ localhost/yii2system/backend/department/index

My Company

Alexander Pierce  
Online

Search...

ข้อมูลหลัก

</> ข้อมูลแผนก

ข้อมูลตำแหน่ง

ข้อมูลผู้ใช้

Menu Yii2

แผนก ทั้งหมด 35 รายการ

รหัส	แผนก	ลบ
02	ฝ่ายการเงินและพัสดุ2	🗑
03	ฝ่ายโภชนาการ2	🗑
04	กลุ่มงานจิตเวชชุมชน2	🗑
05	กลุ่มงานฟื้นฟูสมรรถภาพผู้ป่วยจิตเวช2	🗑
06	กลุ่มงานจิตเวชทางเลือกและแพทย์แผนไทย2	🗑
07	กลุ่มงานการแพทย์2	🗑

- เมทอด where() ใช้เมื่อเราต้องการกรองข้อมูล สำหรับเงื่อนไขต่างๆ เช่น ต้องการ select ข้อมูลเฉพาะ รหัสแผนกเท่ากับ 02 เท่านั้น ตัวอย่าง ลองบันทึกไฟล์ แล้วทดสอบรันดูอีกครั้ง <http://localhost/yii2system/backend/department/index>

\$id = '02';

\$department = \$query->select(['department\_id','department\_name'])->from('department')->where(['department\_id' => \$id])->all();

```
public function actionIndex() {
    $query = new Query();
    $count = $query->from('department')->count();

    $id = '02'; //ตัวแปร $id เราสามารถรับค่าจาก Url มาได้
    $department = $query->select(['department_id','department_name'])
        ->from('department')
        ->where(['department_id' => $id])
        ->all();

    return $this->render('index', [
        'departments' => $department,
        'count' => $count,
    ]);
}
```

หากต้องการทำแบบค้นหาข้อมูลก็สามารถใช้ like ได้เหมือน SQL ปกติครับ เช่น ถ้าต้องการค้นหาชื่อแผนก สามารถเขียนได้ดังนี้

\$search = 'กลุ่ม'; //ค่าค้นหาที่ต้องการ เราสามารถสร้างตัวแปร \$search รับมาจากฟอร์ม หรือ URL ได้

\$department = \$query->from('department')

```
->where([
    'like','department_name',$search
])->all();
```

```
public function actionIndex() {

    $query = new Query();
    $count = $query->from('department')->count();

    $search = 'กลุ่ม';
    $department = $query->from('department')
        ->where([
            'like','department_name',$search
        ])->all();

    return $this->render('index', [
        'departments' => $department,
        'count' => $count,
    ]);
}
```

- เมธอด orderBy() ใช้เมื่อเราต้องการเรียงลำดับข้อมูลจากมากไปน้อย (DESC) หรือจากน้อยไปมาก (ASC) โดยเราใช้ค่าคงที่ในการเรียงลำดับ คือ SORT\_ASC และ SORT\_DESC ถ้าต้องการเรียงลำดับหลายคอลัมน์ก็เพิ่ม array ได้อีก 1 ชุด ตัวอย่าง

```
$department = $query->from('department')
    ->orderBy([
        'department_name' => SORT_DESC,
    ])->all();
```

```
public function actionIndex() {

    $query = new Query();
    $count = $query->from('department')->count();

    $department = $query->from('department')
        ->orderBy([
            'department_name' => SORT_DESC,
        ])->all();

    return $this->render('index', [
        'departments' => $department,
        'count' => $count,
    ]);
}
```



- เมธอด sum(), average(), max(), min() ก็เทียบได้กับเป็นฟังก์ชัน SQL ปกติ เพื่อหาผลรวม ค่าเฉลี่ย ค่ามากที่สุด และค่าน้อยที่สุดนั่นเอง ตัวอย่างเช่น ผมต้องการหาค่าสูงสุดของรหัสแผนก ก็ให้ระบุคอลัมน์รหัสแผนกเข้าไปใน method max() ดังนี้

```
$max = $query->from('department')->max('department_id');
```

ส่วนเมธอดอื่นๆ ก็ใช้ในลักษณะเดียวกันโดยเปลี่ยนจาก max เป็น sum เป็นต้น

- เมธอด join() ใช้เมื่อต้องการ join ตารางที่มีความสัมพันธ์กัน ได้แก่ LEFT JOIN, INNER JOIN, RIGHT JOIN เป็นต้น เช่น ถ้าผมต้องการ join โดยใช้ LEFT JOIN ก็ให้ระบุเป็น argument ตัวแรก ดังนี้

```
$query->join('LEFT JOIN', 'department', 'person.department_id = department.department_id')->all();
```

นอกจากนี้เรายังสามารถระบุ method เหล่านี้แทน join ได้เช่น leftJoin(), innerJoin(), rightJoin() เป็นต้น ตัวอย่างเช่น

```
$model = $query->select('department_name')
->from('person')
->innerJoin('department', 'person.department_id = department.department_id')->all();
```

**สรุป** Query Builder เหมาะกับการคิวรีข้อมูลที่มีความซับซ้อน เช่น การทำรายงานระบบ ข้อดีคือ รองรับทุกฐานข้อมูล แต่เราก็ต้องเขียนในเชิง Programming นั่นเอง

## การใช้งาน Active Record

Active Record เป็น object-oriented interface ที่ใช้สำหรับเข้าถึง และจัดการข้อมูลในฐานข้อมูลซึ่งเราไม่ต้องเปลืองแรงในการเขียน SQL และประหยัดเวลาในการพัฒนามากที่สุดครับ แต่โดยพื้นฐานแล้วเรายังต้องเรียกใช้ method บางอย่างเช่นเดียวกับ Query Builder โดยส่วนตัวแล้วผมแนะนำว่า ให้ใช้ Active Record เป็นหลักในระบบเราเลยครับ ถ้ามันซับซ้อนขึ้นเช่น พวกคิวรีรายงาน ค่อยใช้ Query Builder หรือ DAO มาลองเปรียบเทียบ Active Record กับ DAO ครับ ว่าทำไมผมถึงแนะนำ เช่น ถ้าเราต้องการเพิ่มข้อมูล 1 รายการเข้าไปในตารางฐานข้อมูล ถ้าเป็น DAO เราอาจเขียนได้ดังนี้

```
$db->createCommand('INSERT INTO customer ('name') VALUES (:name)',[':name' => 'CodingThailand',])->execute();
```

แต่ถ้าเราเขียน Active Record จะเขียนได้แบบนี้

```
$customer = new Customer();
```

```
$customer->name = 'CodingThailand';
```

```
$customer->save();
```

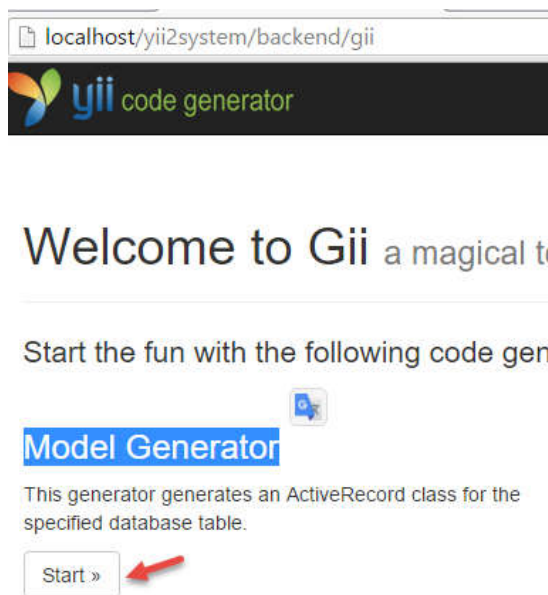
จะสังเกตเห็นว่าเขียนง่ายกว่ากันมาก! และรองรับทุกฐานข้อมูลด้วยนะ

การใช้ Active Record นั้น เราต้องสร้าง Models ก่อน โดยสามารถใช้ Gii สร้างโค้ดให้เราได้ จากนั้นก็ค่อยสืบทอดคุณสมบัติ (extends) yii\db\ActiveRecord แค่นี้ก็เรียบร้อยแล้ว (ปกติ Gii จะสร้างให้แล้ว) คลาส yii\db\ActiveRecord จะสืบทอดคุณสมบัติมาจาก yii\base\Model อีกที ทำให้เราจะได้คุณสมบัติและความสามารถของ Models มาด้วย เช่น attributes, validation rules เป็น มาทดลองสร้าง Models กันครับ

## การสร้าง Models ด้วย Gii เพื่อใช้งานกับ Active Record

ในหัวข้อนี้เราจะสร้าง Model โดยใช้ตัวอย่างจากตาราง คำนานหน้าชื่อ (prefix) กันครับ มีขั้นตอนดังนี้

1. เข้า Url พิมพ์ <http://localhost/yii2system/backend/gii> เพื่อเข้าใช้งาน Gii มองหา Model Generator คลิกที่ปุ่ม Start เพื่อเริ่มสร้าง Model



2. กรอกรายละเอียดต่างๆ เพื่อสร้าง Model (ใช้ตาราง prefix เป็นตัวอย่าง)

**Note:** ปกติในส่วนการกรอก Model Class ชื่อ Model จะขึ้นต้นด้วยตัวพิมพ์ใหญ่เสมอ และแน่นอนในส่วนการกรอก Namespace ถ้าต้องการสร้างในส่วน frontend ก็ให้เปลี่ยนจาก backend\models เป็น frontend\models ครับ

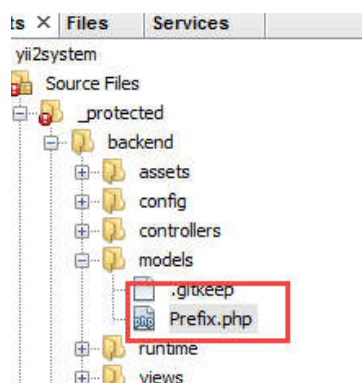
# Model Generator

This generator generates an ActiveRecord class for the specified database table.

<b>Table Name</b>	<input type="text" value="prefix"/>
<b>Model Class</b>	<input type="text" value="Prefix"/>
<b>Namespace</b>	<input type="text" value="backend\models"/>
<b>Base Class</b>	<input type="text" value="yii\db\ActiveRecord"/>
<b>Database Connection ID</b>	<input type="text" value="db"/>
<input type="checkbox"/> Use Table Prefix	
<input checked="" type="checkbox"/> Generate Relations	

จากนั้น ให้คลิกปุ่ม Preview -> แล้วคลิกปุ่ม Generate Gii จะสร้างไฟล์และโค้ดให้เรา สามารถตรวจสอบได้ที่

backend/models/Prefix.php แค่นี้ก็เสร็จเรียบร้อยแล้วครับ



เปิดไฟล์ Prefix.php จะเห็นได้ว่า คลาส Prefix จะสืบทอดคุณสมบัติจาก yii\db\ActiveRecord แค่นี้เราก็พร้อมใช้ Active Record กันแล้วครับ ที่นี้มาดูโค้ดที่ Gii สร้างให้เราครับว่า แต่ละส่วนมีอะไรบ้าง ดังนี้

```
<?php
```

```
namespace backend\models;
```

```
use Yii;
```

```
/**
```

```
 * This is the model class for table "prefix".
```

```

*

* @property string $prefix_id
* @property string $prefix_name
*

* @property Person[] $people

*/

class Prefix extends \yii\db\ActiveRecord //สืบทอดคุณสมบัติจากคลาส ActiveRecord
{
    /**
     * @inheritdoc
     */

    public static function tableName() //กำหนดชื่อตาราง
    {
        return 'prefix';
    }

    /**
     * @inheritdoc
     */

    public function rules() //กฎเพื่อตรวจสอบความถูกต้องของข้อมูล เช่น เมื่อเพิ่มข้อมูลต้องกรอกข้อมูลให้ครบ และถูกต้อง
    {
        return [
            [['prefix_id'], 'required'],
            [['prefix_id'], 'string', 'max' => 2],
            [['prefix_name'], 'string', 'max' => 50]
        ];
    }

    /**
     * @inheritdoc
     */

    public function attributeLabels() //กำหนด Label ของคอลัมน์เพื่อแสดงคำอธิบายของคอลัมน์ต่างๆ เช่น ที่ฟอร์ม เป็นต้น

```

```

{
    return [
        'prefix_id' => 'Prefix ID',
        'prefix_name' => 'Prefix Name',
    ];
}

/**
 * @return \yii\db\ActiveQuery
 */
public function getPeople() //ชื่อ Relation ใช้เมื่อต้องการ join ตาราง โดยการใช้เราจะตัด get ออกเหลือแค่ People เท่านั้น
{
    return $this->hasMany(Person::className(), ['prefix_id' => 'prefix_id']);
}
}

```

## ขั้นตอนการควิรีข้อมูลด้วย Active Record

ในการควิรีข้อมูลด้วย Active Record จะมีขั้นตอน 3 ขั้นตอน ดังนี้

- 1.สร้าง query object โดยเรียกเมธอด **find()** (ถ้าเทียบกับ query builder คือ **select()** นั่นเอง)
- 2.เรียกใช้ query building methods ต่างๆ ก็เหมือนกับ **Query Builder** ได้แก่ **from()**, **where()**, **orderBy()** เป็นต้น
- 3.เรียกใช้ query method เพื่อรับข้อมูล เช่น **all()**, **one()**, **count()** เป็นต้น

**สรุปคือ** การใช้งาน Active Record นั้นจะคล้ายกับ Query Builder นั่นเอง แต่เปลี่ยนจาก **select()** มาเป็น **find()** ครบ แต่ข้อดีของ Active Record อีกอย่างคือ คือ การเพิ่ม และแก้ไขข้อมูลทำได้ง่ายกว่า เพียงแค่เรียกใช้ method **save()** เท่านั้น

## ตัวอย่างการควิรีข้อมูล

// ผลลัพธ์จะแสดงข้อมูลลูกค้า ที่มีรหัสลูกค้า เท่ากับ 123

//เทียบกับคำสั่ง SQL ปกติจะเป็น **SELECT \* FROM `customer` WHERE `id` = 123**

```
$customer = Customer::find()
```

```
->where(['id' => 123])
```

```
->one();
```

// ผลลัพธ์จะแสดงข้อมูลลูกค้าทั้งหมด โดยมีเงื่อนไข status เท่ากับ 1 และเรียงลำดับตามรหัสลูกค้าจากน้อยไปมาก

// เทียบกับ SQL ปกติ คือ SELECT \* FROM `customer` WHERE `status` = '1' ORDER BY `id`

```
$customers = Customer::find()
```

```
->where(['status' => '1'])
```

```
->orderBy('id')
```

```
->all();
```

// ผลลัพธ์ นับจำนวนลูกค้าที่มีสถานะเท่ากับ 1

// เทียบกับ SQL ปกติ คือ SELECT COUNT(\*) FROM `customer` WHERE `status` = 1

```
$count = Customer::find()
```

```
->where(['status' => '1'])
```

```
->count();
```

## การคิวรีข้อมูลด้วย Active Record แบบวิธีลัด (shortcut methods)

การคิวรีโดยใช้ shortcut methods โค้ดจะสั้น และสะดวกกว่าวิธีที่ผ่านมา มีอยู่ 2 เมธอด ได้แก่

- findOne() จะคืนค่าผลลัพธ์แถวแรกของการคิวรี
- findAll() จะคืนค่าผลลัพธ์ทั้งหมดของการคิวรี

ตัวอย่างการใช้งาน รูปแบบอื่นๆ

// ผลลัพธ์จะแสดงข้อมูลลูกค้า ที่มีรหัสลูกค้า เท่ากับ 123

//เทียบกับคำสั่ง SQL ปกติจะเป็น SELECT \* FROM `customer` WHERE `id` = 123

```
$customer = Customer::findOne(123);
```

// ผลลัพธ์จะแสดงข้อมูลลูกค้าที่ประกอบด้วยรหัส 100, 101, 123 หรือ 124

////เทียบกับคำสั่ง SQL ปกติ คือ SELECT \* FROM `customer` WHERE `id` IN (100, 101, 123, 124)

```
$customers = Customer::findAll([100, 101, 123, 124]);
```

```
// ผลลัพธ์จะแสดงข้อมูลลูกค้าที่รหัส เท่ากับ 123 และ สถานะ เท่ากับ 1

// เทียบกับคำสั่ง SQL ปกติ คือ SELECT * FROM `customer` WHERE `id` = 123 AND `status` = '1'

$customer = Customer::findOne([

    'id' => 123,

    'status' => '1',

]);

// แสดงข้อมูลลูกค้า โดยมีเงื่อนไขสถานะ เท่ากับ 0

// เทียบกับคำสั่ง SQL ปกติ คือ SELECT * FROM `customer` WHERE `status` = '0'

$customers = Customer::findAll([

    'status' => '0',

]);
```

## การเข้าถึง Attributes ข้อมูล เมื่อใช้ Active Record

การเรียกใช้ หรือเข้าถึง attributes หรือเรียกอีกอย่างหนึ่งว่า คอลัมน์ในตารางฐานข้อมูลนั้น เราสามารถอ้างถึงโดยใช้ instance ที่สร้างขึ้น แล้วตามด้วยชื่อคอลัมน์ในตารางได้เลย ตัวอย่างเช่น หากเราต้องการใช้งาน attributes id และ email ก็เรียกใช้ได้ดังนี้

```
// "id" และ "email" คือ ชื่อคอลัมน์ของตาราง "customer" นั่นเอง

$customer = Customer::findOne(123);

$id = $customer->id;

$email = $customer->email;
```

## การสร้าง Attributes ใหม่ อาจใช้คำนวณ หรือจัดรูปแบบข้อมูล เมื่อใช้ Active Record

บางครั้งเราอยากสร้าง attributes ขึ้นมาใหม่เพื่อใช้ในการเปลี่ยนรูปข้อมูล หรือจัดรูปแบบข้อมูล เรียกว่า data transformation methods การเขียนคือ ให้กำหนด set หรือ get แล้วตามด้วยชื่อ attribute ที่เราต้องการครับ เช่น

```
class Customer extends ActiveRecord

{

    // ...
```

```
public function getBirthdayText()
{
    return date('Y/m/d', $this->birthday);
}
```

```
public function setBirthdayText($value)
{
    $this->birthday = strtotime($value);
}
}
```

จากโค้ดจะเห็นว่า เราสร้าง method `getBirthdayText()` และ `setBirthdayText($value)` ขึ้นมา (get เอาไว้ดึงค่าข้อมูล และ set เอาไว้กำหนดค่าข้อมูล) หากเราต้องการเรียกใช้ method สองตัวนี้ในรูปแบบ attribute ก็สามารถเรียกใช้ ได้ดังนี้

- `$customer->birthdayText;` (ตัด get ออกไปนั่นเอง) ในทำนองเดียวกันถ้าจะ set ค่าก็ตัด set ออกไปครับ

## เทคนิคการใช้ `asArray()` กับ Active Record

ในกรณีที่เราจำเป็นต้องเรียกดูข้อมูลปริมาณมาก เพื่อประสิทธิภาพที่ดี แนะนำให้ใช้เมธอด `asArray()` ครับ ตัวอย่าง

```
$prefix = Prefix::find()->as Array()->all();
```

## การเพิ่ม หรือแก้ไขข้อมูลด้วย Active Record

การเพิ่มข้อมูล หรือแก้ไขข้อมูล นั้นง่ายมากๆ แค่เรียกใช้ method `save()` ครับ ดังนี้

ตัวอย่าง การเพิ่มข้อมูล

```
$prefix = new Prefix();
$prefix->prefix_id = '20';
$prefix->prefix_name = 'นาย';
$prefix->save();
```

ตัวอย่าง การแก้ไขข้อมูล แน่นอนเราต้องคิดว่าเราจะแก้ไขข้อมูลแถวไหนก่อน ค่อย `save()` อีกที

```
$prefix = Prefix::findOne(20);
$prefix->prefix_name = 'นางสาว';
```



```
$prefix->save();
```

## การเพิ่มหรือลดค่าของคอลัมน์ (Updating Counters)

เราสามารถบวกเพิ่ม หรือลบค่าที่เฉพาะเจาะจงก็ได้กับคอลัมน์ที่มีประเภทข้อมูลเป็นตัวเลขครับ เช่น ลดค่าคงเหลือ สินค้า 10 ชิ้นเมื่อมีการซื้อสินค้า, เพิ่มค่าจำนวนการเข้าชมเว็บไซต์ทีละ 1 เป็นต้น ตัวอย่าง

```
$product = Product::findOne(123);  
$product->updateCounters(['product_stock' => 1]);
```

จากโค้ดด้านบน ถ้าเทียบกับ SQL แล้ว คือ UPDATE `product` SET `product\_stock` = `product\_stock` + 1 WHERE `id` = 123  
หากต้องการลดค่าก็ให้ติดเครื่องหมายลบเข้าไปครับ

## การลบข้อมูล โดยใช้ Active Record

การลบข้อมูล เราสามารถลบข้อมูลได้ทีละแถว หรือลบข้อมูลทั้งหมดก็ได้ ประกอบด้วย

- ใช้ delete() เพื่อลบข้อมูลทีละแถว แต่ก่อนลบก็ต้องควมรู้ข้อมูลแถวที่ต้องการลบก่อน ตัวอย่างเช่น  

```
$customer = Customer::findOne(123);  
$customer->delete();
```
- ใช้ deleteAll() เพื่อลบข้อมูลทั้งหมดในตาราง แต่ให้ระวังด้วยครับ หรือเราอาจมีเงื่อนไขใส่เข้าไปด้วยก็ได้ ตัวอย่างเช่น  

```
Customer::deleteAll(['status' => '0']);
```

## การใช้ Transactions

ตัวอย่างโค้ด

```
$customer = Customer::findOne(123);
```

```
Customer::getDb()->transaction(function($db) use ($customer) {  
    $customer->id = 200;  
    $customer->save();  
    // ...สามารถใส่คำสั่งอื่นๆได้ตรงนี้...
```

```
});
```

// หรืออีก 1 วิธี ใช้ตามนี้

```
$transaction = Customer::getDb()->beginTransaction();
```

```
try {
```

```
    $customer->id = 200;
```

```
    $customer->save();
```

```
    // ...ใส่คำสั่งการทำงานอื่นๆตรงนี้...
```

```
    $transaction->commit(); //สั่ง commit ข้อมูล
```

```
} catch(\Exception $e) {
```

```
    $transaction->rollBack(); //สั่ง rollback ข้อมูล
```

```
    throw $e;
```

```
}
```

## การ Join ตาราง โดยใช้ Active Record

การ join ตารางที่มีความสัมพันธ์กัน นั้นมีหลายวิธี แต่ผมจะยกตัวอย่างวิธีที่สั้นกระชับให้ครับ โดยใช้ `joinWith()` นั่นเอง  
ตัวอย่าง เช่น

```
$query = Person::find()->joinWith(['prefix','position','department']);
```

จากตัวอย่างโค้ด เราให้ `Person` join ทั้งหมด 3 ตาราง โดยแต่ละตารางค้นด้วยเครื่องหมายคอมม่า นั่นเอง ง่ายมาก! แต่อย่าลืมว่าเราต้อง  
สร้าง FK (foreign key) ระหว่างตารางให้เรียบร้อยก่อนใช้ Gii สร้างโค้ดอัตโนมัติให้ด้วยนะครับ

## การใช้ Data Providers สำหรับแบ่งหน้า และจัดเรียงข้อมูล

Data Providers มีหน้าที่สำหรับรับข้อมูลที่เกิดจากการคิวรีมาแบ่งหน้า หรือจะจัดเรียงข้อมูล โดยสามารถใช้ได้ทั้ง DAO, Query Builder  
หรือ Active Record ก็ได้ และยังสามารถนับจำนวน record ทั้งหมด หรือ record เฉพาะในหน้าที่แบ่ง นั้นๆได้ด้วย การใช้ Data Providers  
นั้นส่วนใหญ่แล้วเราจะใช้ร่วมกับ GridView ครับ

Data Providers ประกอบด้วย

- SqlDataProvider: ใช้กับ DAO นั้นเองจะคืนค่าข้อมูลในฐานะข้อมูลเป็นรูปแบบ Array
- ActiveRecord: ใช้ได้ทั้งกับ Query Builder และ Active Record
- ArrayDataProvider: ใช้กับข้อมูล Array ขนาดใหญ่

**Note:** เราไม่ควรใช้ Array Data Provider ในกรณีที่ต้องการโหลดข้อมูลเยอะๆ เช่น 10,000 แถว เป็นต้น เพราะ Array Data Provider จะโหลดข้อมูลมาทั้งหมดก่อนค่อยมาแบ่งหน้าหรือเรียงลำดับ แนะนำควรไปใช้ ActiveRecord แทน

## การใช้ SqlDataProvider สำหรับแบ่งหน้า และจัดเรียงข้อมูล

ปกติแล้ว SqlDataProvider จะใช้กับ DAO ครบ แต่มีเงื่อนไขว่าเราต้อง count หรือนับจำนวน record ทั้งหมด ส่งเข้าไปด้วย และต้อง use yii\data\SqlDataProvider; ด้วยครับ ลองเปิดไฟล์ PositionController.php แล้วให้สิทธิ์ในการเข้าถึง actions grid ด้วยโดยเพิ่มสมาชิก arrays เข้าไปอีก 1 ชุด ใน method behaviors ดังนี้

```
public function behaviors() {  
    return [  
        'access' => [  
            'class' => AccessControl::className(),  
            'rules' => [  
                [  
                    'actions' => ['login', 'error'],  
                    'allow' => true,  
                ],  
                [  
                    'actions' => ['logout', 'index', 'delete','grid'],  
                    'allow' => true,  
                    'roles' => ['@'],  
                ],  
            ],  
        ],  
        'verbs' => [  
            'class' => VerbFilter::className(),
```

```

        'actions' => [
            'logout' => ['post'],
        ],
    ],
];
}

```

จากนั้นเพิ่มโค้ด use yii\data\SqlDataProvider; ไว้ด้านบน ต่อด้วยลองเพิ่ม method actionGrid() ดังนี้ครับ

```

public function actionGrid() {
    $count = Yii::$app->db->createCommand('SELECT COUNT(*) FROM position')->queryScalar();

    $dataProvider = new SqlDataProvider([
        'sql' => 'SELECT * FROM position', //คำสั่ง SQL
        'totalCount' => $count, //จำนวนแถวทั้งหมด
        'pagination' => [
            'pageSize' => 10, //จะแบ่งหน้าละกี่แถว
        ],
    ]);

    return $this->render('grid', [
        'dataProvider' => $dataProvider,
    ]);
}

```

ในส่วนของ view ให้สร้างไฟล์ backed/views/position/grid.php เราจะใช้ GridView ในการแสดงผล ดังนี้

```

<?php
$this->title = Yii::t('app', 'ตำแหน่ง');
$this->params['breadcrumbs'][] = $this->title;
?>
<p>
<?= \yii\grid\GridView::widget([
    'dataProvider' => $dataProvider,

```

```
'columns' => [
    'position_id',
    'position_name',
],
]) ?>
</p>
```

เสร็จแล้ว ทดสอบโดยพิมพ์ Url ดังนี้ <http://localhost/yii2system/backend/position/grid>

Position Id	Position Name
20	ช่างไฟฟ้า
17	ช่างไม้
99	ช่างไม้ครุภัณฑ์
44	ทันตแพทย์
94	นักการภารโรง
40	นักการแพทย์แผนไทย
64	นักกิจกรรมบำบัด
28	นักจัดการงานทั่วไป
08	นักจัดการงานทั่วไปชำนาญการพิเศษ
55	นักจิตวิทยา

## การใช้งาน GridView

GridView เป็น widgets ที่ช่วยให้เราสามารถแสดงผลข้อมูลต่างๆในรูปแบบของ table (grid) ได้ โดยอย่างน้อยต้องใส่พารามิเตอร์ 2 ตัว ได้แก่ dataProvider และ columns ครับ โดยในคอลัมน์ต่างๆ เราสามารถจัดรูปแบบข้อมูลได้ เช่น การเงิน, Url, Email เป็นต้น

```
<?= \yii\grid\GridView::widget([
    'dataProvider' => $dataProvider,
```

```
'columns' => [
  'id',
  'product_name',
  'product_date:datetime',
  'product_price:currency',
],
]) ?>
```

เราสามารถ custom คอลัมน์ใน grid ได้ โดยระบุเป็นชุดของ array เข้าไปได้แก่

attribute	ระบุชื่อ attribute ของ model
format	จัดรูปแบบการแสดงผล
header	ส่วนหัวของคอลัมน์นั้น
footer	ส่วนท้ายของคอลัมน์นั้น
visible	กำหนด true หรือ false เพื่อกำหนดว่าแสดงหรือไม่แสดงข้อมูล
content	callback สำหรับแสดงผลข้อมูลออกไป
value	ค่าข้อมูล attribute ของ model

## การกรองข้อมูลใน GridView (filters)

การกรองข้อมูล โดยปกติแล้ว เราจะเห็นแถวใหม่เป็นกล่อง (text inputs) สำหรับกรองข้อมูลด้านบนของตาราง ซึ่งจะมีการสร้างมาให้มาแล้วนั่นเอง

และถ้าเราอยากกรองข้อมูลในรูปแบบ Dropdown list ก็เพียงแค่ส่งค่าข้อมูลที่เป็น array เข้าไปใน property filter หรือจะดึงค่ามาจากรฐานข้อมูลก็ได้เช่นเดียวกัน ตัวอย่าง

```
[
  'attribute' => 'prefix',
  'value' => 'prefix.prefix_name',
  // 'filter' => ['นาย', 'นาง', 'นางสาว'], // ใส่เป็น array แบบนี้ก็ได้
```

'filter' => \backend\models\Person::getPrefixName(), //ดึงมาจากรฐานข้อมูลผ่าน model Person

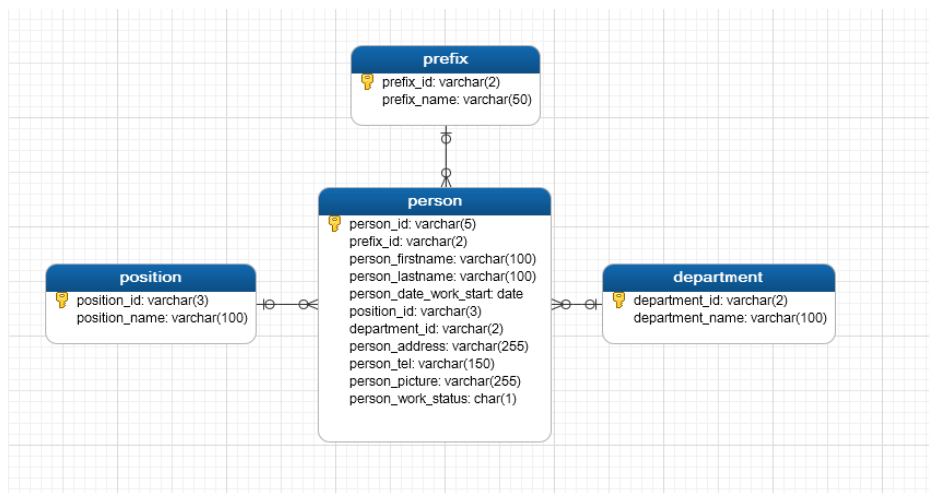
],

#	Person ID	Prefix	Person Firstname
	<input type="text"/>	<input type="text"/>	
1	001	นาย	พรินทร์
2	004	นางสาว	มานิด
3	005	ว่าที่ร้อยตรี	จ่าเอก
4	006	เด็กชาย	ภูทัก
5	007	เด็กหญิง	ร้อยตำรวจเอก
6	008	พลตรี	บุญขลิ
		พลโท	รียอร์
		พลเอก	
		สิบตำรวจตรี	สุระเชษฐ์

## Workshop: การจัดการข้อมูลบุคลากร (Person)

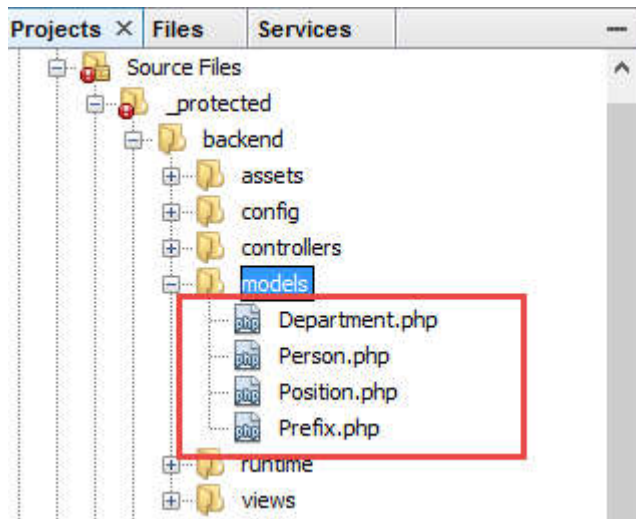
หลังจากที่เราได้เรียนรู้ Active Record การ Join ตาราง และการใช้งาน GridView กันแล้ว มาลองใช้ความรู้ทั้งหมดมาสร้าง

Workshop ดูครับ ในภาพรวมของตารางฐานข้อมูลแต่ละตารางมีความสัมพันธ์กัน ดังนี้



จากภาพ ตาราง person ถ้าเราอยากได้ข้อมูลค่านำหน้า (prefix) ข้อมูลตำแหน่ง (position) ข้อมูลหน่วยงาน (department) เราต้อง join ตาราง 3 ตาราง

**Step 1:** ใช้ Gii สร้าง Models ทั้งหมดประกอบด้วย prefix (สร้างแล้ว), position, department และ person โดยขั้นตอนเหมือนการสร้างโมเดล Prefix ครับ เมื่อสร้างเสร็จแล้วเราจะได้ models ทั้งหมด ดังนี้



Step 2: ใช้ Gii สร้างระบบ CRUD (Create, Read, Update, Delete) ให้กับ person มีขั้นตอน ดังนี้

1) กรอกรายละเอียดให้กับ CRUD Generator ดังนี้

## CRUD Generator

This generator generates a controller and views that implement CRUD (Create, Read, Update, Delete) data model.

### Model Class

backend\models\Person

### Search Model Class

backend\models\PersonSearch

### Controller Class

backend\controllers\PersonController

### View Path

### Base Controller Class

yii\web\Controller

### Widget Used in Index Page

GridView

☐ Enable I18N



2) คลิกปุ่ม Preview -> แล้วคลิกปุ่ม Generate เป็นอันเสร็จเรียบร้อย

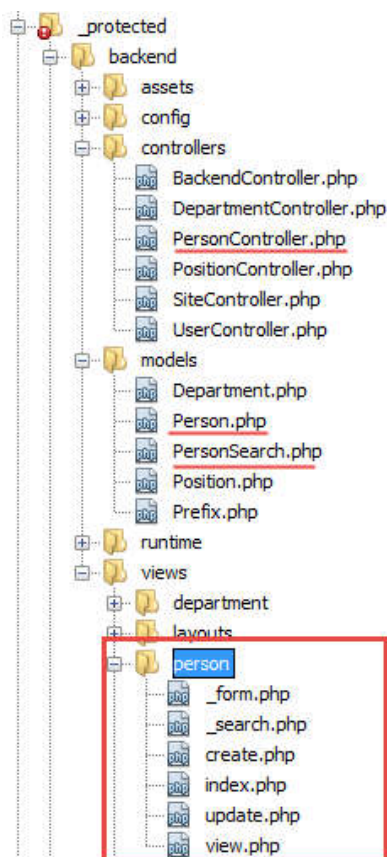
Preview Generate

Click on the above **Generate** button to generate the files selected below:

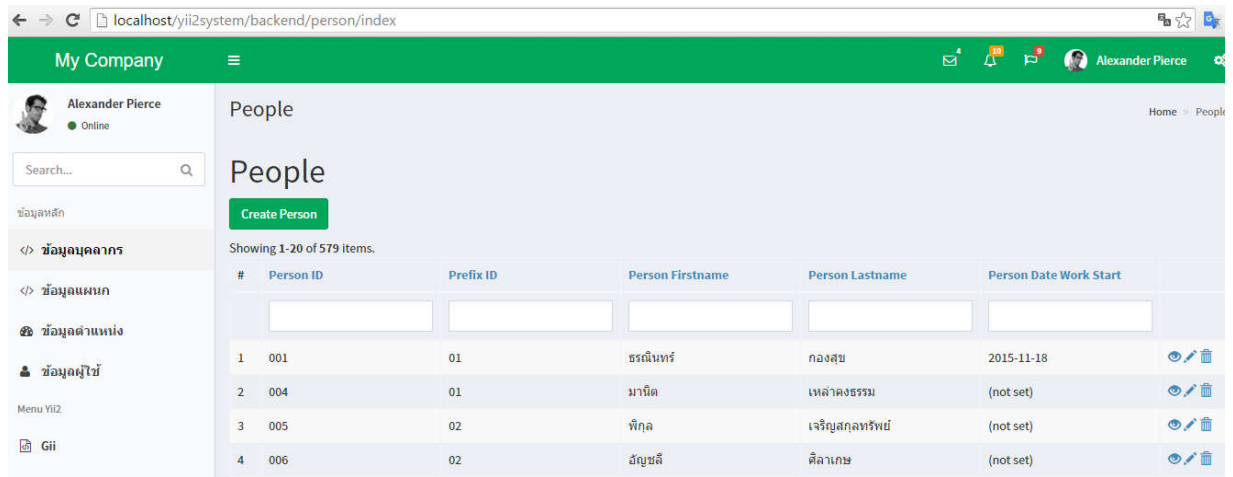
☒ Create ☒ Unchanged ☒ Overwrite

Code File	Action	<input checked="" type="checkbox"/>
controllers\PersonController.php	create	<input checked="" type="checkbox"/>
models\PersonSearch.php	create	<input checked="" type="checkbox"/>
views\person\_form.php	create	<input checked="" type="checkbox"/>
views\person\_search.php	create	<input checked="" type="checkbox"/>
views\person\create.php	create	<input checked="" type="checkbox"/>
views\person\index.php	create	<input checked="" type="checkbox"/>
views\person\update.php	create	<input checked="" type="checkbox"/>
views\person\view.php	create	<input checked="" type="checkbox"/>

โดย Gii จะสร้างไฟล์ให้เราดังต่อไปนี้ครับ



ทดลองรันดู หรืออาจสร้างเมนูด้านซ้ายเพิ่มก็ได้ <http://localhost/yii2system/backend/person/index>



3) เมื่อรันแล้วเราจะเห็นว่าข้อมูลใน GridView จะแสดงข้อมูลเฉพาะตาราง person เท่านั้น หากเราต้องการ join ตารางเพื่อแสดง คำนำหน้า แสดงตำแหน่ง หรือแสดงหน่วยงาน ก็ต้อง join table ก่อนครับ โดยให้เปิดไฟล์ backend/models/PersonSearch.php เพิ่มโค้ดทั้งหมด ดังต่อไปนี้

```
<?php
```

```
namespace backend\models;
```

```
use Yii;
```

```
use yii\base\Model;
```

```
use yii\data\ActiveDataProvider;
```

```
use backend\models\Person;
```

```
/**
```

```
 * PersonSearch represents the model behind the search form about `backend\models\Person`.
```

```
 */
```

```
class PersonSearch extends Person
```

```
{
```

```
    public $prefix;
```

```
    public $position;
```

```
    public $department;
```

```
/**
```

```

* @inheritdoc
*/

public function rules()
{
    return [

        [['person_id', 'prefix_id', 'person_firstname', 'person_lastname', 'person_date_work_start', 'position_id',
'department_id', 'person_address', 'person_tel', 'person_picture', 'person_work_status','prefix','position','department'], 'safe'],

    ];
}

/**
 * @inheritdoc
 */
public function scenarios()
{
    // bypass scenarios() implementation in the parent class
    return Model::scenarios();
}

/**
 * Creates data provider instance with search query applied
 *
 * @param array $params
 *
 * @return ActiveDataProvider
 */
public function search($params)
{
    $query = Person::find()->joinWith(['prefix','position','department']);

    $dataProvider = new ActiveDataProvider([
        'query' => $query,
    ]);
}

```

```
]);
```

```
$dataProvider->sort->attributes['position'] = [  
    'asc' => ['position.position_name' => SORT_ASC],  
    'desc' => ['position.position_name' => SORT_DESC],  
];
```

```
$dataProvider->sort->attributes['department'] = [  
    'asc' => ['department.department_name' => SORT_ASC],  
    'desc' => ['department.department_name' => SORT_DESC],  
];
```

```
$this->load($params);
```

```
if (!$this->validate()) {  
    // uncomment the following line if you do not want to return any records when validation fails  
    // $query->where('0=1');  
    return $dataProvider;  
}
```

```
$query->andFilterWhere([  
    'person_date_work_start' => $this->person_date_work_start,  
]);
```

```
$query->andFilterWhere(['like', 'person_id', $this->person_id])  
->andFilterWhere(['like', 'prefix_id', $this->prefix_id])  
->andFilterWhere(['like', 'person_firstname', $this->person_firstname])  
->andFilterWhere(['like', 'person_lastname', $this->person_lastname])  
->andFilterWhere(['like', 'position_id', $this->position_id])  
->andFilterWhere(['like', 'department_id', $this->department_id])  
->andFilterWhere(['like', 'person_address', $this->person_address])  
->andFilterWhere(['like', 'person_tel', $this->person_tel])
```

```

->andFilterWhere(['like', 'person_picture', $this->person_picture])
->andFilterWhere(['like', 'person_work_status', $this->person_work_status])
->andFilterWhere(['like', 'prefix.prefix_name', $this->prefix])
->andFilterWhere(['like', 'position.position_name', $this->position])
->andFilterWhere(['like', 'department.department_name', $this->department]);

return $dataProvider;
}
}

```

อธิบายโค้ดแต่ละส่วน ไฟล์ PersonSearch.php จะเป็นการคิวรีข้อมูลเพื่อสำหรับแบ่งหน้า join table และไว้ค้นหาข้อมูลต่างๆ

```

public $prefix;

public $position;

public $department;

```

Attributes 2 ตัว ด้านบนนี้เราประกาศไว้เพื่อเก็บค่าข้อมูลสำหรับการค้นหาคำนำหน้า ตำแหน่ง แผนก ตามลำดับ

```

$query = Person::find()->joinWith(['prefix','position','department']);

```

โค้ดด้านบนนี้เอาไว้คิวรีข้อมูลทั้งหมดโดย join table 3 ตารางได้แก่ prefix, position, department

```

$dataProvider->sort->attributes['position'] = [
    'asc' => ['position.position_name' => SORT_ASC],
    'desc' => ['position.position_name' => SORT_DESC],
];

$dataProvider->sort->attributes['department'] = [
    'asc' => ['department.department_name' => SORT_ASC],
    'desc' => ['department.department_name' => SORT_DESC],
];

```

โค้ดชุดนี้เป็นการกำหนดการเรียงลำดับให้กับ attribute ตำแหน่ง และ attribute แผนก ให้กับ GridView โดยสามารถเรียงได้ทั้งจากมากไปหาน้อย และจากน้อยไปหามาก

```
->andFilterWhere(['like', 'prefix.prefix_name', $this->prefix])  
  
->andFilterWhere(['like', 'position.position_name', $this->position])  
  
->andFilterWhere(['like', 'department.department_name', $this->department]);
```

โค้ดชุดนี้ มีไว้ให้เราสามารถกรองข้อมูลค่านำหน้า ตำแหน่ง แผนก ให้กับ GridView ได้ (สังเกตจากการใช้ like)

**Step 4:** ก่อนปรับแต่ง GridView ถ้าเราอยากกรองข้อมูลในรูปแบบ DropDownList ให้เปิดไฟล์ backend/models/Person.php แล้วให้เพิ่ม static method อีก 1 method ดังนี้

```
public static function getPrefixName()  
{  
    $prefixName = Prefix::find()->asArray()->all();  
    return \yii\helpers\ArrayHelper::map($prefixName, 'prefix_name', 'prefix_name');  
}
```

โค้ดส่วนนี้จะเราดึงข้อมูลค่านำหน้าขึ้นมาในรูปแบบ Array แล้วทำการ map เพื่อให้ใช้กับ DropDownList ได้

**Step 5:** ปรับแต่ง GridView ให้เปิดไฟล์ backend/views/person/index.php โดยในขั้นตอนนี้เราจะเพิ่มความสามารถให้ GridView ให้สามารถใช้งาน AJAX ได้ด้วย เพิ่มโค้ด ดังนี้ครับ

```
<?php  
  
use yii\helpers\Html;  
  
use yii\grid\GridView;  
  
use yii\widgets\Pjax;  
  
$this->title = 'บุคลากร';  
  
$this->params['breadcrumbs'][] = $this->title;  
  
?>
```

```
<div class="person-index">
```

```
<p>
```

```
<?= Html::a('สร้างใหม่', ['create'], ['class' => 'btn btn-success']) ?>
```

```
</p>
```

```
<?php Pjax::begin([ 'enablePushState' => false ]); ?>
```

```
<?= GridView::widget([
```

```
    'dataProvider' => $dataProvider,
```

```
    'filterModel' => $searchModel,
```

```
    'columns' => [
```

```
        ['class' => 'yii\grid\SerialColumn'],
```

```
        //'person_id',
```

```
        [
```

```
            'attribute' => 'person_id',
```

```
            'value' => 'person_id',
```

```
            'contentOptions' => ['class' => 'text-center'],
```

```
            'headerOptions' => ['class' => 'text-center'],
```

```
        ],
```

```
        //'prefix_id',
```

```
        //prefix.prefix_name',
```

```
        [
```

```
            'attribute' => 'prefix',
```

```
            'value' => 'prefix.prefix_name',
```

```
            //'filter' => ['นาย','นาง','นางสาว'],
```

```
            //'filter' => \backend\models\Person::getPrefixName(),
```

```
            'filter' => Html::activeDropDownList($searchModel, 'prefix', \backend\models\Person::getPrefixName()
```

```
, ['class' => 'form-control', 'prompt' => '--กรุณาเลือกกรายการ--']),
```

```
        ],
```

```
        'person_firstname',
```

```
        'person_lastname',
```

```

        //'position.position_name',

        [

            'attribute' => 'position',

            'value' => 'position.position_name',

        ],

        //'department.department_name',

        [

            'attribute' => 'department',

            'value' => 'department.department_name',

        ],

        //'person_date_work_start',

        // 'position_id',

        // 'department_id',

        // 'person_address',

        // 'person_tel',

        // 'person_picture',

        // 'person_work_status',

        [

            'attribute' => 'person_work_status',

            'content' => function($model, $key, $index, $column) {

                return $model->person_work_status == 0 ? '<i class="fa fa-check"></i>' : '<i class="fa fa-close"></i>';

            },

            /*'value' => function($model, $key, $index, $column) {

                $status = "";

                if ($model->person_work_status == 0) {

                    $status = "ทำงานอยู่";

                } else if ($model->person_work_status == 1) {

                    $status = "ลาออก";

                } else {

                    $status = "ลาพักผ่อน";

                }

            }

```



```

        return $status;
    }*/
],
[
    'class' => 'yii\grid\ActionColumn',
    'header' => 'เครื่องมือ',
    'headerOptions' => ['width' => '100'],
    'template' => '{view} {update} {delete}',
    'buttons' => [
        'view' => function($url, $model) {
            return Html::a('<span class="glyphicon glyphicon-search"></span>', $url, ['title'=>'ดูรายละเอียด']);
        },
        'delete' => function($url, $model) {
            return Html::a('<span class="glyphicon glyphicon-remove"></span>', $url, ['title' => 'ลบรายการ', 'data' =>
                ['confirm' => 'แน่ใจว่าต้องการลบรายการนี้?', 'method' => 'post']]);
        }
    ],
],
],

],
]);?>

<?php Pjax::end();?>

</div>

```

บุคลากร Home > บุคลากร

สร้างใหม่

Showing 1-20 of 579 items.

#	Person ID	Prefix	Person Firstname	Person Lastname	Position	Department	Person Work Status	เครื่องมือ
	<input type="text"/>	--กรุณาเลือก--	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	
1	001	นาย	ธรัตน์	ทองสุข	(not set)	ฝ่ายการเงินและพัสดุ2	✖	<a href="#">🔍</a> <a href="#">✎</a> <a href="#">✖</a>
2	004	นาย	มานิต	เหล้าคงธรรม	(not set)	ฝ่ายโภชนาการ2	✖	<a href="#">🔍</a> <a href="#">✎</a> <a href="#">✖</a>
3	005	นาง	พิกุล	เจริญสกุลทรัพย์	(not set)	ฝ่ายโภชนาการ2	✓	<a href="#">🔍</a> <a href="#">✎</a> <a href="#">✖</a>
4	006	นาง	อัญชลี	ศิลาเกษ	(not set)	ฝ่ายโภชนาการ2	✖	<a href="#">🔍</a> <a href="#">✎</a> <a href="#">✖</a>
5	007	นาง	วิริยธร	จุมพระบุตร	(not set)	ฝ่ายโภชนาการ2	✓	<a href="#">🔍</a> <a href="#">✎</a> <a href="#">✖</a>
6	008	นาย	สุรเชษฐ์	เรียงบาล	(not set)	กลุ่มงานจิตเวชชุมชน2	✖	<a href="#">🔍</a> <a href="#">✎</a> <a href="#">✖</a>
7	009	นาง	สำเภา	เงินหมื่น	(not set)	กลุ่มงานจิตเวชชุมชน2	✓	<a href="#">🔍</a> <a href="#">✎</a> <a href="#">✖</a>

## อธิบายเพิ่มเติมในส่วน GridView ดังนี้

หากลองรันดูหน้าเว็บเพจในส่วน GridView ถ้าเราค้นข้อมูลใน Text inputs หรือลองคลิกที่ตัวแบ่งหน้า จะพบว่า GridView จะทำงานในรูปแบบของ AJAX เรียบร้อย โดยการใช้งาน AJAX นั้นให้เราเพิ่มโค้ด ดังนี้

- use yii\widgets\Pjax;
- พิมพ์โค้ด <?php Pjax::begin([ 'enablePushState' => false ]); ?> ในส่วนบนของ Grid
- พิมพ์โค้ด <?php Pjax::end(); ?> ในส่วนท้ายของ Grid

```
[  
    'attribute' => 'person_work_status',  
    'content' => function($model, $key, $index, $column) {  
        return $model->person_work_status == 0 ? '<i class="fa fa-check"></i>' : '<i class="fa fa-close"></i>';  
    },  
]
```

โค้ดในส่วนนี้เราจะแสดงไอคอนเพื่อแสดงสถานะของบุคลากรโดยตรวจสอบว่า person\_work\_status ถ้าเท่ากับ 0 จึงให้แสดงไอคอน check นอกนั้นให้แสดงไอคอน close นั่นเอง

```
[  
    'class' => 'yii\grid\ActionColumn',  
    'header' => 'เครื่องมือ',  
    'headerOptions' => ['width' => '100'],  
    'template' => '{view} {update} {delete}',  
    'buttons' => [  
        'view' => function($url, $model) {  
            return Html::a('<span class="glyphicon glyphicon-search"></span>', $url, ['title' => 'ดูรายละเอียด']);  
        },  
        'delete' => function($url, $model) {  
            return Html::a('<span class="glyphicon glyphicon-remove"></span>', $url, ['title' => 'ลบรายการ', 'data' =>  
                ['confirm' => 'แน่ใจว่าต้องการลบรายการนี้?', 'method' => 'post']]);  
        },  
    ],  
],
```

โค้ดส่วนนี้เป็นส่วนของเครื่องมือของ GridView โดยมีปุ่มเครื่องมือ 3 ตัว ได้แก่ view, update, delete แต่เราปรับแต่งใส่ไอคอนใหม่ให้กับ view และ delete ใหม่ด้วย

## บทที่ 3: การสร้างรายงานด้วย Yii 2

### สร้างรายงานในรูปแบบ PDF และรองรับฟอนต์ภาษาไทย ด้วย mPDF

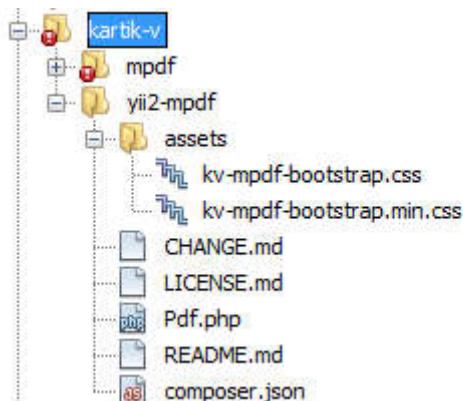
ในหัวข้อนี้เราจะสร้างรายงานในรูปแบบ PDF ครับ โดยเราจะใช้ yii2-mpdf extension (krajee-v) สามารถดูรายละเอียดการติดตั้งและใช้งานแบบเต็มๆได้ที่ <http://demos.krajee.com/site/mpdf> ส่วนรายละเอียดของ mPDF จะอยู่ที่ <http://www.mpdf1.com/mpdf/index.php>

#### ขั้นตอนการติดตั้ง yii2-mpdf ของ kartik-v

1.เปิดไฟล์ composer.json แล้วเพิ่ม "kartik-v/yii2-mpdf": "\*" การติดตั้งให้เราสั่ง `composer update` เหมือนกับที่เราเคยติดตั้ง AdminLTE

```
"require": {  
    "php": ">=5.4.0",  
    "yiisoft/yii2": "*",  
    "yiisoft/yii2-bootstrap": "*",  
    "yiisoft/yii2-swiftmailer": "*",  
    "nenad/yii2-password-strength": "*",  
    "mihailedev/yii2-ckeditor": "*",  
    "dmstr/yii2-adminlte-asset": "2.*",  
    "kartik-v/yii2-mpdf": "*"   
},
```

2.เมื่อติดตั้งเสร็จแล้ว ให้ตรวจสอบก่อนว่า library ต่างๆมาครบหรือไม่ โดยตรวจสอบได้ที่โฟลเดอร์ `_protected/vendor/kartik-v/`



3.ติดตั้งฟอนต์ภาษาไทย ในที่นี้ผมจะยกตัวอย่างโดยใช้ฟอนต์ THSarabunNew การดาวน์โหลดฟอนต์นี้อาจค้น google หรือเข้าตามนี้ก็ได้ ครับ <http://www.f0nt.com/release/th-sarabun-new/> เมื่อดาวน์โหลดมาแล้วให้แตก zip ไฟล์เตรียมไว้ได้เลย

4. จากนั้น Copy ฟอนต์ทั้งหมด ได้แก่ THSarabunNew.ttf, THSarabunNew Italic.ttf, THSarabunNew BoldItalic.ttf, THSarabunNew Bold.ttf ไปวาง (paste) ที่โฟลเดอร์ protected/vender/kartik-v/mpdf/ttfonts/

5. เปิดไฟล์ protected/vender/kartik-v/mpdf/config\_fonts.php เลื่อนลงมาบรรทัดที่ 137 เพิ่ม array อีก 1 ชุด ดังนี้

```
137         "freemono" => array(  
138             'R' => "FreeMono.ttf",  
139             'B' => "FreeMonoBold.ttf",  
140             'I' => "FreeMonoOblique.ttf",  
141             'BI' => "FreeMonoBoldOblique.ttf",  
142         ),  
143  
144         "thsarabunnew" => array(  
145             'R' => "THSarabunNew.ttf",  
146             'B' => "THSarabunNew Bold.ttf",  
147             'I' => "THSarabunNew Italic.ttf",  
148             'BI' => "THSarabunNew BoldItalic.ttf",  
149         ),  
150
```

6. เปิดไฟล์ protected/vender/kartik-v/yii2-pdf/assets/kv-mpdf-bootstrap.css เลื่อนลงมาบรรทัดที่ 100 ในส่วนของ body โดยให้กำหนดฟอนต์ให้เป็น THSarabunNew ดังนี้

```
100 body {  
101     font-family: "THSarabunNew";  
102     font-size: 14px;  
103     line-height: 1.42857143;  
104     color: #333;  
105     background-color: #fff;  
106 }
```

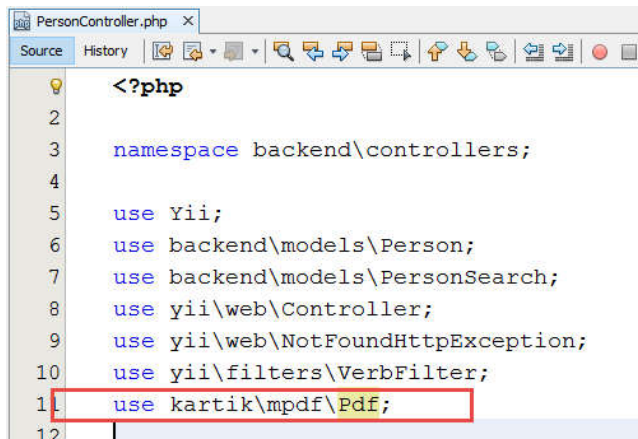
7. เมื่อตั้งค่าเรียบร้อยแล้ว มาทดลองสร้างรายงานด้วย pdf กันครับ โดยหลักการในการสร้าง pdf นั้นเราจะเรียกใช้คลาส pdf ที่ controller ตามปกติ และสร้าง view เพื่อทำเป็นหน้ารายงานนั่นเอง โดย yii2-mpdf มีการตั้งค่าที่สำคัญๆ ดังนี้

- mode** กำหนดรูปแบบการเข้ารหัสภาษาเพื่อการแสดงผล เช่น utf-8 เป็นต้น
- format** กำหนดประเภทหรือขนาดเอกสารที่จะแสดงผล เช่น Pdf::FORMAT\_A4 หรือ Pdf::FORMAT\_A3 เป็นต้น
- cssFile** เราสามารถกำหนดตำแหน่งของไฟล์ .css เพื่อใช้สำหรับจัดรูปแบบเอกสารได้
- content** กำหนดเนื้อหาเอกสาร HTML ที่จะแปลงเป็นเอกสาร PDF ในที่นี้ก็คือ ไฟล์ view นั่นเอง
- filename** กำหนดชื่อไฟล์ให้กับเอกสาร PDF

**destination** กำหนดที่อยู่ปลายทางเมื่อสร้างเอกสาร PDF เสร็จแล้วว่าจะให้อยู่ในรูปแบบใด เช่น Pdf::DEST\_BROWSER ก็ให้แสดงที่ browser หรือ Pdf::DEST\_DOWNLOAD ให้ดาวน์โหลดไฟล์เลย

**method** กำหนดรายละเอียดอื่นๆ โดยระบุ method/function ที่มีใน mPDF มาตรฐาน เช่น การกำหนด header หรือ footer

8. เปิดไฟล์ backend/controllers/PersonController.php เสร็จแล้วให้ use kartik\mpdf\Pdf; ด้านบนเพื่อเตรียมเรียกใช้คลาส Pdf



จากนั้นให้ สร้าง actionPDF โดยเพิ่มโค้ด ดังนี้

```
public function actionPdf() {
```

```
$query = Person::find()->limit(50); //ตรงนี้สามารถควิ่ข้อมูลได้ตามใจชอบ
```

```
$dataProvider = new \yii\data\ActiveDataProvider([
```

```
    'query' => $query,
```

```
    'pagination' => false, //เพื่อการแสดงผลทั้งหมด เลยปิดไม่ให้แบ่งหน้า
```

```
]);
```

```
$content = $this->renderPartial('report',[
```

```
    'dataProvider' => $dataProvider,
```

```
]);
```

```
// setup kartik\mpdf\Pdf component
```

```
$pdf = new Pdf([
```

```
    // set to use core fonts only
```

```
    'mode' => Pdf::MODE_UTF8,
```

```
    // A4 paper format
```

```

'format' => Pdf::FORMAT_A4,

// 'format' => [100,200], // กว้าง,สูง สำหรับ custom ขนาดกระดาษเอง

// filename

'filename' => time(),

// portrait orientation

'orientation' => Pdf::ORIENT_PORTRAIT,

// stream to browser inline

'destination' => Pdf::DEST_BROWSER,

// your html content input

'content' => $content,

// format content from your own css file if needed or use the

// enhanced bootstrap css built by Krajee for mPDF formatting

'cssFile' => '@vendor/kartik-v/yii2-mpdf/assets/kv-mpdf-bootstrap.css',

// any css to be embedded if required

'cssInline' => '.kv-heading-1{font-size:18px}',

// set mPDF properties on the fly

'options' => [

    'title' => 'codingthailand',

    'subject' => 'person report',

    'keywords' => 'person,codingthailand',

],

// call mPDF methods on the fly

'methods' => [

    'SetHeader'=>['รายงานโดย codingthailand || ออกรายงานเมื่อ: '.Yii::$app->formatter->asDatetime(time())],

    'SetFooter'=>['หน้าที่ {PAGENO}'],

]

]);

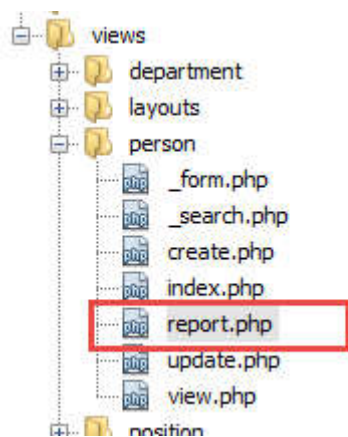
// return the pdf output as per the destination setting

return $pdf->render();

}

```

9.สร้าง view สำหรับรายงาน โดยการสร้างไฟล์ใหม่ขึ้นมาตั้งชื่อว่า report.php ให้สร้างไว้ที่ backend/views/person/



เปิดไฟล์ report.php เพิ่มโค้ดดังนี้

```
<?php
```

```
use yii\grid\GridView;
```

```
use yii\helpers\Url;
```

```
?>
```

```
<div class="person-index">
```

```
<div class="text-center">
```

```

```

```
</div>
```

```
<h1 class="text-center">รายงานบุคลากร ทั้งหมดจำนวน <?= $dataProvider->getTotalCount(); ?> คน</h1>
```

```
<?= GridView::widget([
```

```
'dataProvider' => $dataProvider,
```

```
'columns' => [
```

```
['class' => 'yii\grid\SerialColumn'],
```

```
[
```

```
'attribute' => 'person_id',
```

```
'value' => 'person_id',
```

```
'contentOptions' => ['class' => 'text-center'],
```

```
'headerOptions' => ['class' => 'text-center'],
```

```
],
```

```

        'person_firstname',
        'person_lastname',
    ],
    ]; ?>
</div>

```

หากต้องการเพิ่มรูปภาพเข้ามายังไฟล์ สามารถ copy ไฟล์รูปภาพ ไว้ที่โฟลเดอร์ uploads (อยู่นอกสุด) ได้ ในที่นี้ผมทดลองโดยการ copy ไฟล์ชื่อว่า logo.jpg ไปวางไว้ครับ ตามนี้

```

<div class="text-center">
    
</div>

```

การนับจำนวน record ที่ส่งมาจาก dataProvider นั้น เราเรียกใช้ method getTotalCount() ได้เลย ดังนี้

```

<h1 class="text-center">รายงานบุคลากร ทั้งหมดจำนวน <?= $dataProvider->getTotalCount(); ?> คน</h1>

```

10.บันทึกไฟล์ทั้งหมดแล้วลองรัน Url <http://localhost/yii2system/backend/person/pdf> เพื่อเรียกรายงานครับ หรืออยากทำเป็นปุ่ม หรือลิงก์ก็ได้เช่นเดียวกัน

localhost/yii2system/backend/person/pdf

land 1 / 3

รายงานโดย codingthailand

ออกรายงานเมื่อ: 31/12/2015 21:39:01



## รายงานบุคลากร ทั้งหมดจำนวน 50 คน

Total 50 items.

#	Person ID	Person Firstname	Person Lastname
1	004	มานิต	เหล่าคงธรรม
2	005	พิกุล	เจริญสกุลทรัพย์
3	006	อัญชลี	ศิลาเกษ
4	007	วิริยอร์	จุมพระบุตร
5	008	สุรเชษฐ์	เวียงบาล



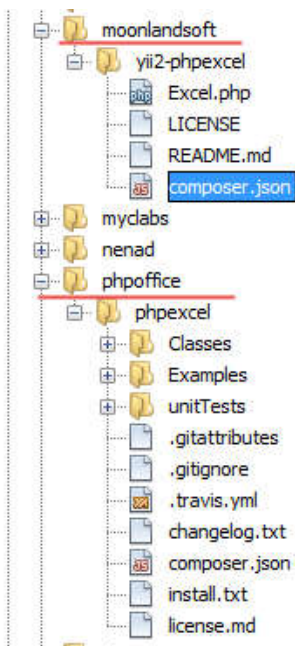
## การส่งออกและนำเข้าไฟล์ Excel

การส่งออก (Export) และการนำเข้า (Import) ไฟล์ Excel เราจะใช้ extension ชื่อว่า moonlandsoft/yii2-phpexcel ครับ มีขั้นตอนการติดตั้ง ดังนี้

1. เปิดไฟล์ composer.json เพิ่มบรรทัด "moonlandsoft/yii2-phpexcel": "\*" ไว้ล่างสุด เสร็จแล้วสั่ง `composer update` เหมือนเดิมครับ

```
"require": {  
    "php": ">=5.4.0",  
    "yiisoft/yii2": "*",  
    "yiisoft/yii2-bootstrap": "*",  
    "yiisoft/yii2-swiftmailer": "*",  
    "nenad/yii2-password-strength": "*",  
    "mihaildev/yii2-ckeditor": "*",  
    "dmstr/yii2-adminlte-asset": "2.*",  
    "kartik-v/yii2-mpdf": "*",  
    "moonlandsoft/yii2-phpexcel": "*"  
},
```

2. ตรวจสอบในโฟลเดอร์ `_protected/vendor/moonlandsoft` และ `_protected/vendor/phpoffice` ว่าไฟล์มาครบหรือไม่



3. ในตัวอย่างนี้เราจะทดลองส่งออกไฟล์ Excel ข้อมูลตาราง position ครับ ให้เปิดไฟล์ `backend/controllers/PositionController.php` แล้วให้เพิ่มสิทธิ์ สำหรับ action export ในส่วนของ `behaviors()` ดังนี้

```
public function behaviors() {  
  
    return [  
  
        'access' => [  
  
            'class' => AccessControl::className(),
```

```

'rules' => [
    [
        'actions' => ['login', 'error'],
        'allow' => true,
    ],
    [
        'actions' => ['logout', 'index', 'delete', 'grid', 'export'],
        'allow' => true,
        'roles' => ['@'],
    ],
],
],
'verbs' => [
    'class' => VerbFilter::className(),
    'actions' => [
        'logout' => ['post'],
    ],
],
];
}

```

จากนั้นให้เพิ่มโค้ด method actionExport() ดังนี้

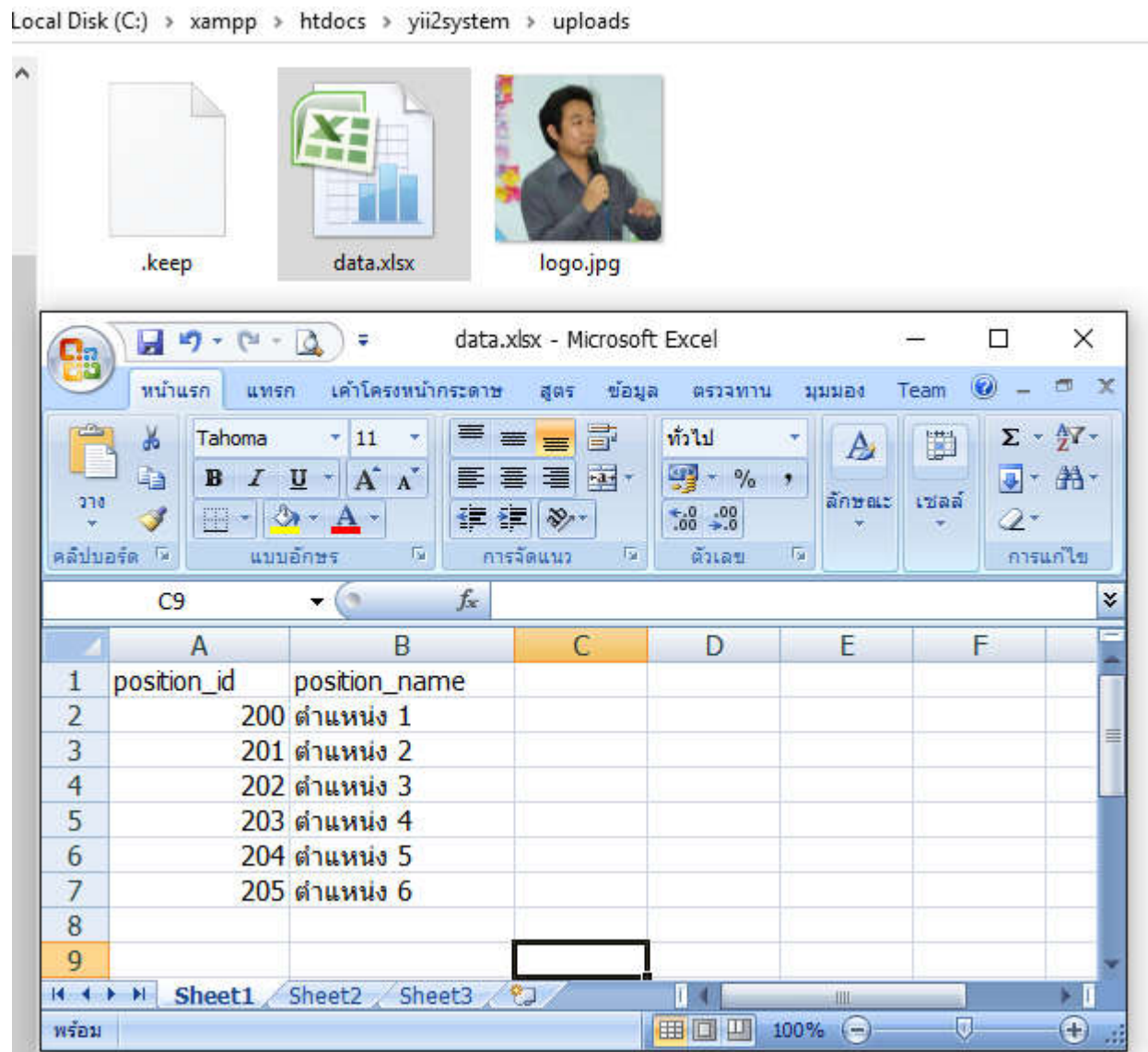
```

public function actionExport() {
    $model = Position::find()->all(); //ในส่วนนี้เราสามารถดึงข้อมูลอย่างไรก็ได้ครับ ถ้าเราต้องการ
    return \moonland\phpexcel\Excel::widget([
        'models' => $model,
        'mode' => 'export', //default value as 'export'
        'fileName' => time(),
    ]);
}

```

4.ทดลองรัน Url <http://localhost/yii2system/backend/position/export> แค่นี้เราก็จะสามารถส่งออกไฟล์ Excel ได้เรียบร้อยแล้ว

5.การนำเข้าไฟล์ Excel ให้เราเตรียมไฟล์ excel สำหรับ import ให้เรียบร้อย ในที่นี้ผมจะสร้างไฟล์ชื่อว่า data.xlsx ไว้ที่โฟลเดอร์ uploads ตัวอย่าง เช่น



7.เพิ่มสิทธิ์ action import ใน method behaviors() ดังนี้

```
public function behaviors() {  
    return [  
        'access' => [  
            'class' => AccessControl::className(),  
            'rules' => [  
                [  
                    'actions' => ['login', 'error'],  
                    'allow' => true,  
                ],  
            ],  
        ],  
    ];  
}
```

```

    ],
    [
        'actions' => ['logout', 'index', 'delete','grid','export','import'],
        'allow' => true,
        'roles' => ['@'],
    ],
],
],
],
'verbs' => [
    'class' => VerbFilter::className(),
    'actions' => [
        'logout' => ['post'],
    ],
],
];
}

```

#### 8.เพิ่มโค้ด actionImport() ดังนี้

```

public function actionImport() {
    $data = \moonland\phpexcel\Excel::widget([
        'mode' => 'import',
        'fileName' => 'C:\xampp\htdocs\yii2system\uploads\data.xlsx', //path จริงที่อยูไฟล์
        //setFirstRecordAsKeys' => true, // if you want to set the keys of record column with first record, if it not set, the
header with use the alphabet column on excel.

        // 'setIndexSheetByName' => true, // set this if your excel data with multiple worksheet, the index of array will be set
with the sheet name. If this not set, the index will use numeric.

        'getOnlySheet' => 'Sheet1', // you can set this property if you want to get the specified sheet from the excel data with
multiple worksheet.
    ]);

    //echo print_r($data);

```

```
//insert data array to table

foreach ($data as $key => $value) {

    Yii::$app->db->createCommand()->insert('position', [

        'position_id' => $value['position_id'],

        'position_name' => $value['position_name'],

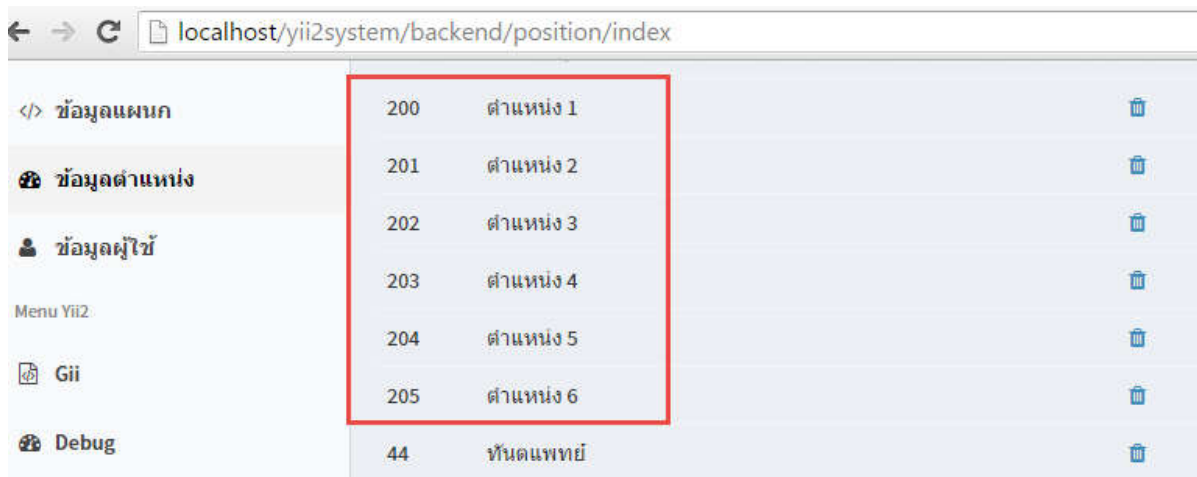
    ])->execute();

}

return $this->redirect(['index']);

}
```

9.ส่งรัน Url เพื่อนำเข้าข้อมูล <http://localhost/yii2system/backend/position/import> แล้วตรวจสอบข้อมูลใหม่ที่เราเข้ามาครับ



ID	ตำแหน่ง	ลบ
200	ตำแหน่ง 1	
201	ตำแหน่ง 2	
202	ตำแหน่ง 3	
203	ตำแหน่ง 4	
204	ตำแหน่ง 5	
205	ตำแหน่ง 6	
44	ทันตแพทย์	

Note: ทั้งในส่วนการ Import และ Export Excel นั้น ดูรายละเอียด options ต่างๆทั้งหมดได้ที่

<https://github.com/moonlandsoft/yii2-phpexcel>

## สร้างรายงานในรูปแบบ Chart ด้วย HighCharts

การสร้างรายงานในรูปแบบ Chart มีหลากหลายรูปแบบ ในเล่มนี้ผมจะแนะนำรูปแบบที่เราใช้งานบ่อยๆ ได้แก่ Bar Charts, Line Charts และ Pie Charts ครับ โดยเราจะใช้ extension ของ miloschuman/yii2-highcharts-widget มีขั้นตอนการติดตั้ง และใช้งาน ดังนี้

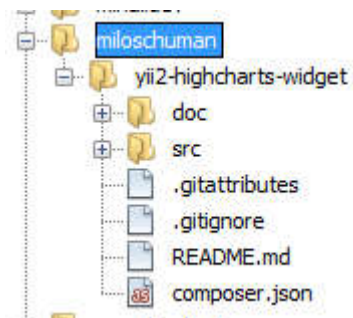
1.เปิดไฟล์ composer.json เพิ่มโค้ด "miloschuman/yii2-highcharts-widget": "dev-master" แล้วสั่ง `composer update`

```

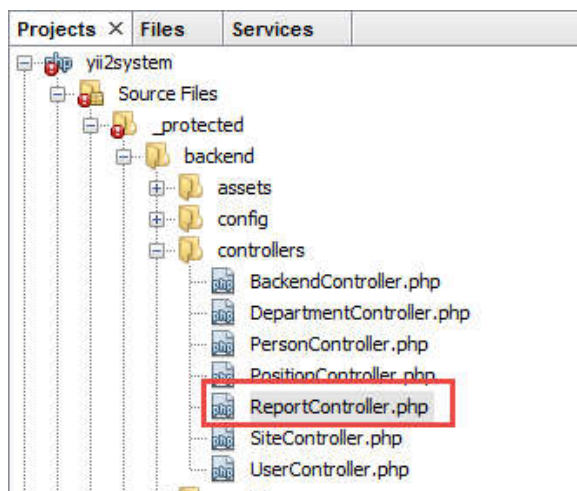
"require": {
    "php": ">=5.4.0",
    "yiisoft/yii2": "*",
    "yiisoft/yii2-bootstrap": "*",
    "yiisoft/yii2-swiftmailer": "*",
    "nenad/yii2-password-strength": "*",
    "mihaildev/yii2-ckeditor": "*",
    "dmstr/yii2-adminlte-asset": "2.*",
    "kartik-v/yii2-mpdf": "*",
    "moonlandsoft/yii2-phpexcel": "*",
    "miloschuman/yii2-highcharts-widget": "dev-master"
},

```

2. ตรวจสอบไฟล์ว่าติดตั้งเรียบร้อยแล้วหรือไม่ ที่ `_protected/vender/miloschuman`



3. สร้างไฟล์ใหม่ `backend/controllers/ReportController.php` เพื่อใช้กับรายงานแบบ Charts โดยเฉพาะ



เขียนโค้ด ทั้งหมด ดังนี้

```
<?php
```

```
namespace backend\controllers;
```

```
use Yii;
```

```

use yii\web\Controller;

use yii\web\NotFoundHttpException;

use yii\filters\VerbFilter;

use yii\db\Query;

/**
 * PrefixController implements the CRUD actions for Prefix model.
 */
class ReportController extends Controller
{
    public function behaviors()
    {
        return [
            'verbs' => [
                'class' => VerbFilter::className(),
                'actions' => [
                    'delete' => ['post'],
                ],
            ],
        ];
    }

    /**
     * Lists all Prefix models.
     *
     * @return mixed
     */
    public function actionReport1()
    {
        /**
         SELECT
         department.department_name
         FROM

```

```
person
```

```
INNER JOIN department ON person.department_id = department.department_id
```

```
GROUP BY
```

```
department.department_name
```

```
*/
```

```
$query = new Query();
```

```
$department = $query->select('department_name')
```

```
->from('person')
```

```
->innerJoin('department','person.department_id = department.department_id')
```

```
->groupBy('department.department_name')
```

```
->all();
```

```
//Count(person.person_id) AS countperson
```

```
$query2 = new Query();
```

```
$count = $query2->select('Count(person.person_id) AS countperson')
```

```
->from('person')
```

```
->innerJoin('department','person.department_id = department.department_id')
```

```
->groupBy('department.department_name')
```

```
->all();
```

```
return $this->render('report1',[
```

```
    'department' => $department,
```

```
    'count' => $count,
```

```
]);
```

```
}
```

```
public function actionReport2()
```

```
{
```

```
/*
```

```
SELECT
```

```
department.department_name
```

```
FROM
```



```

        person

        INNER JOIN department ON person.department_id = department.department_id

        GROUP BY

        department.department_name

    */

$query = new Query();

$department = $query->select('department_name')

    ->from('person')

    ->innerJoin('department','person.department_id = department.department_id')

    ->groupBy('department.department_name')

    ->all();

//Count(person.person_id) AS countperson

$query2 = new Query();

$count = $query2->select('Count(person.person_id) AS countperson')

    ->from('person')

    ->innerJoin('department','person.department_id = department.department_id')

    ->groupBy('department.department_name')

    ->all();

return $this->render('report2',[

    'department' => $department,

    'count' => $count,

]);

}

```

```

public function actionReport3() {

    /*

    SELECT

    position.position_name,

    Count(person.person_id)

```

```

FROM

person

INNER JOIN position ON person.position_id = position.position_id

GROUP BY

position.position_name

*/

$query2 = new Query();

$data = $query2->select(['position_name','count' => 'Count(person.person_id)'])

->from('person')

->innerJoin('position','person.position_id = position.position_id')

->groupBy('position.position_name')

->all();

return $this->render('report3',[

    'data' => $data,

]);

}

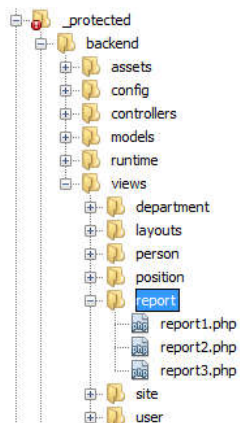
}

```

อธิบายเพิ่มเติม

ในไฟล์ ReportController.php เราจะทำตัวอย่าง Charts ทั้งหมด 3 รูปแบบด้วยกัน โดยทำเป็น action 3 ตัวได้แก่ actionReport1(), actionReport2(), actionReport3() ครบ โดยแต่ละตัวก็จะ render view เพื่อแสดง Chart ในรูปแบบต่างๆ และในส่วน ของ action ต่างๆนั้น ภายในจะเป็นตัวอย่างการควิรข้อมูลรูปแบบต่างๆให้ศึกษา

4.สร้างไฟล์ใหม่ในส่วนของ views ทั้ง 3 ไฟล์ได้แก่ report1.php, report2.php และ report3.php ที่โฟลเดอร์ backend/views/report/



5.เปิดไฟล์ report1.php พิมพ์โค้ด ดังนี้

```
<?php
```

```
use miloschuman\highcharts\Highcharts;
```

```
$this->title = 'รายงาน จำนวนบุคลากรแยกตามแผนก';
```

```
?>
```

```
<div class="row">
```

```
    <div class="col-md-12">
```

```
        <div class="box box-info">
```

```
            <div class="box-header with-border">
```

```
                <h3 class="box-title">Default Box Example</h3>
```

```
                <div class="box-tools pull-right">
```

```
                    <!-- Buttons, labels, and many other things can be placed here! -->
```

```
                    <!-- Here is a label for example -->
```

```
                    <span class="label label-primary">Label</span>
```

```
                </div><!-- /.box-tools -->
```

```
            </div><!-- /.box-header -->
```

```
        <div class="box-body">
```

```
            <?php
```

```
            $cat = array();
```

```
            foreach ($department as $value) {
```

```
                $cat[] = $value['department_name'];
```

```
            }
```

```
            $data = array();
```

```
            foreach ($count as $value) {
```

```
                $data[] = intval($value['countperson']);
```

```
            }
```

```

/*$data = array();

foreach ($count as $value) {

    $data[] = $value['countperson'];

}

$data = array_map('intval', $data);*/

```

```

echo Highcharts::widget([

    'scripts' => [

        'highcharts-3d',

    ],

    'options' => [

        'chart' => [

            'type' => 'column',

            'borderWidth' => 1,

            'borderRadius' => 5,

            'options3d' => [

                'enabled' => true,

                'alpha' => 10,

                'beta' => 30,

            ]

            //'inverted' => true,

        ],

        'title' => ['text' => 'รายงานจำนวนบุคลากร'],

        'subtitle' => ['text' => 'รายงานจำนวนบุคลากรแยกตามแผนก'],

        'credits' => ['enabled' => false],

        'legend' => [

            'align' => 'right',

            'verticalAlign' => 'middle',

            'layout' => 'vertical',

            'borderWidth' => 2,

            'borderRadius' => 3,

        ],

    ],

]);

```

```

        'xAxis' => [
            'title' => ['text' => 'แผนก'],
            'categories' => $cat,
        ],
        'yAxis' => [
            'title' => ['text' => 'จำนวนพนักงาน']
        ],
        'series' => [
            [
                'name' => 'จำนวนพนักงาน',
                'data' => $data,
                'dataLabels' => [
                    'enabled' => true,
                    'x' => -5,
                    'y' => 30,
                ]
            ],
        ],

    ]
    });
?>

</div><!-- /.box-body -->

<div class="box-footer">

    The footer of the box

</div><!-- box-footer -->

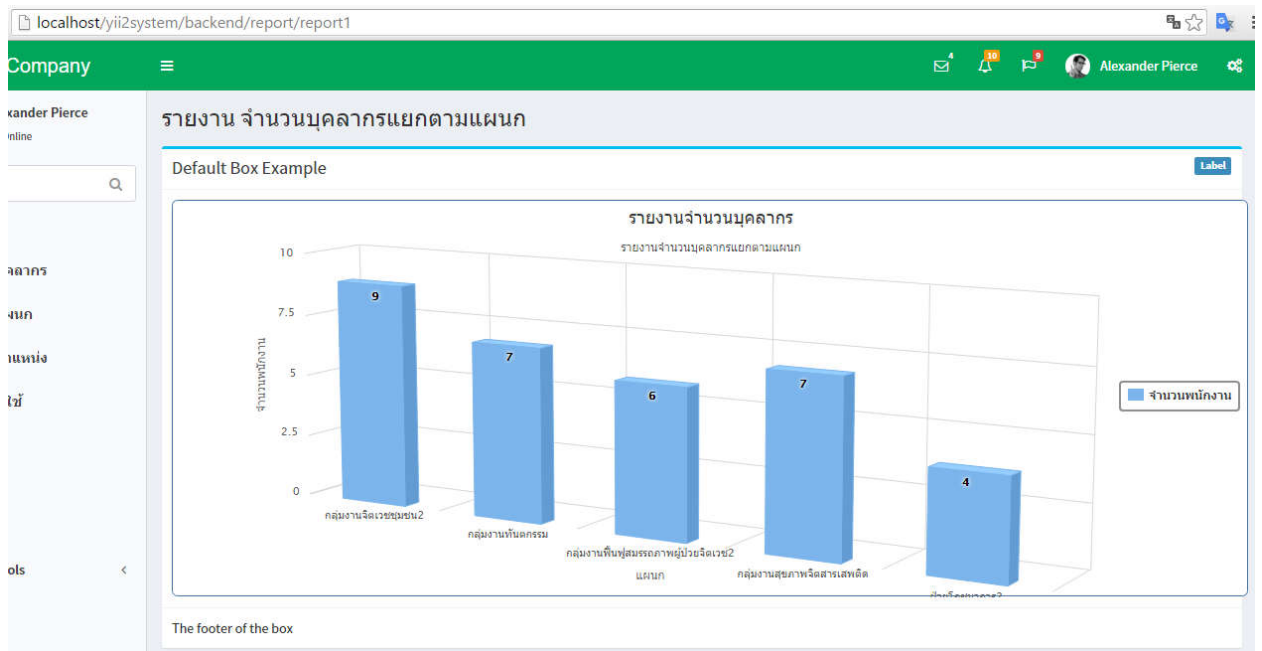
</div><!-- /.box -->

</div>

</div>

```

6.ทดสอบพิมพ์ Url เพื่อแสดงผล Bar Chart <http://localhost/yii2system/backend/report/report1>



7.เปิดไฟล์ report2.php พิมพ์โค้ด ดังนี้

```
<?php
```

```
use miloschuman\highcharts\Highcharts;
```

```
$this->title = 'รายงาน จำนวนบุคลากรแยกตามแผนก';
```

```
?>
```

```
<div class="row">
```

```
<div class="col-md-12">
```

```
<div class="box box-info">
```

```
<div class="box-header with-border">
```

```
<h3 class="box-title">Default Box Example</h3>
```

```
<div class="box-tools pull-right">
```

```
<!-- Buttons, labels, and many other things can be placed here! -->
```

```
<!-- Here is a label for example -->
```

```
<span class="label label-primary">Label</span>
```

```
</div><!-- /.box-tools -->
```

```
</div><!-- /.box-header -->
```

```
<div class="box-body">
```

```
<?php
```

```
//print_r($department);
```

```
//print_r($count);
```

```
$cat = array();
```

```
foreach ($department as $value) {
```

```
    $cat[] = $value['department_name'];
```

```
}
```

```
$data = array();
```

```
foreach ($count as $value) {
```

```
    $data[] = intval($value['countperson']);
```

```
}
```

```
/*$data = array();
```

```
foreach ($count as $value) {
```

```
    $data[] = $value['countperson'];
```

```
}
```

```
$data = array_map('intval', $data);*/
```

```
echo Highcharts::widget([
```

```
    'options' => [
```

```
        'title' => ['text' => 'รายงานจำนวนบุคลากร'],
```

```
        'subtitle' => ['text' => 'รายงานจำนวนบุคลากรแยกตามแผนก'],
```

```
        'credits' => ['enabled' => false],
```

```
        'legend' => [
```

```
            'align' => 'right',
```

```
            'verticalAlign' => 'middle',
```

```
            'layout' => 'vertical',
```

```
            'borderWidth' => 2,
```

```
            'borderRadius' => 3,
```

```
    ],
    'xAxis' => [
      'title' => ['text' => 'แผนก'],
      'categories' => $cat,
    ],
    'yAxis' => [
      'title' => ['text' => 'จำนวนพนักงาน']
    ],
    'series' => [
      [
        'lineWidth' => 5,
        'name' => 'จำนวนพนักงาน',
        'data' => $data,
        'dataLabels' => [
          'enabled' => true,
          'x' => -5,
          'y' => 30,
        ]
      ]
    ],

  ],

]

]);
?>

</div><!-- /.box-body -->

<div class="box-footer">

  The footer of the box

</div><!-- box-footer -->

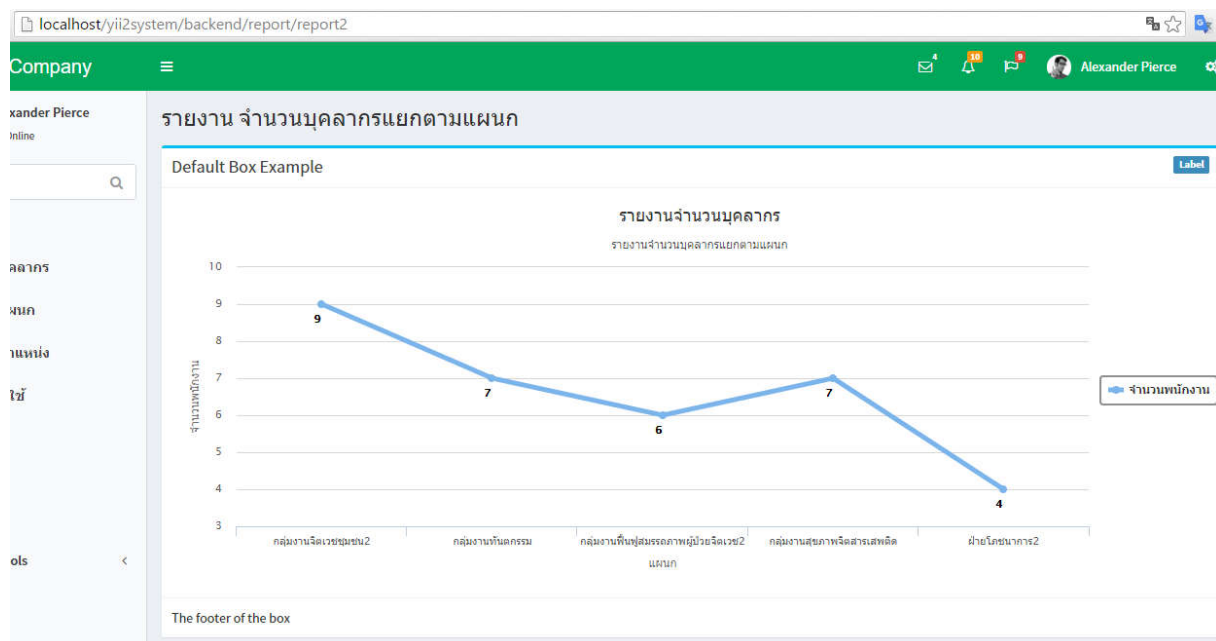
</div><!-- /.box -->

</div>

</div>
```



8.ทดสอบพิมพ์ Url เพื่อแสดงผล Line Chart <http://localhost/yii2system/backend/report/report2>



9.เปิดไฟล์ report3.php พิมพ์โค้ด ดังนี้

```
<?php
```

```
use miloschuman\highcharts\Highcharts;
```

```
use yii\web\JsExpression;
```

```
$this->title = 'รายงาน จำนวนบุคลากรแยกตามตำแหน่ง';
```

```
?>
```

```
<div class="row">
```

```
    <div class="col-md-12">
```

```
        <div class="box box-info">
```

```
            <div class="box-header with-border">
```

```
                <h3 class="box-title">Default Box Example</h3>
```

```
            <div class="box-tools pull-right">
```

```
                <!-- Buttons, labels, and many other things can be placed here! -->
```

```

<!-- Here is a label for example -->

<span class="label label-primary">Label</span>

</div><!-- /.box-tools -->
</div><!-- /.box-header -->
<div class="box-body">

    <?php

        //echo print_r($data);

        $posData = array();
        foreach ($data as $value) {

            extract($value);

            $posData[] = array($value['position_name'], intval($value['count']));
        }

        echo Highcharts::widget([

            'scripts' => [

                'highcharts-3d',

            ],

            'options' => [

                'chart' => [

                    'type' => 'pie',

                    'borderWidth' => 1,

                    'borderRadius' => 5,

                    'options3d' => [

                        'enabled' => true,

                        'alpha' => 55,

                        'beta' => 0,

                    ]

                ],

                'plotOptions' => [

                    'pie' => [

```

```

        'depth' => 50, // 3d
        'showInLegend' => true,
        'dataLabels' => [
            'distance' => -50,
            'style' => [
                'fontWeight' => 'bold',
                'width' => '140px',
            ],
            'formatter' => new JsExpression('function() { return this.point.name + " "+
Highcharts.numberFormat(this.y,0) + " คน" }'),
        ]
    ],
    'title' => ['text' => 'รายงานจำนวนบุคลากร'],
    'subtitle' => ['text' => 'รายงานจำนวนบุคลากรแยกตำแหน่ง'],
    'credits' => ['enabled' => true],
    'legend' => [
        'align' => 'right',
        'verticalAlign' => 'middle',
        'layout' => 'vertical',
        'borderWidth' => 2,
        'borderRadius' => 3,
    ],
    'series' => [
        [
            'name' => 'จำนวนพนักงาน',
            'data' => $posData,
        ],
    ],
],
]

```

```

    });

    ?>

</div><!-- /.box-body -->

<div class="box-footer">

    The footer of the box

</div><!-- box-footer -->

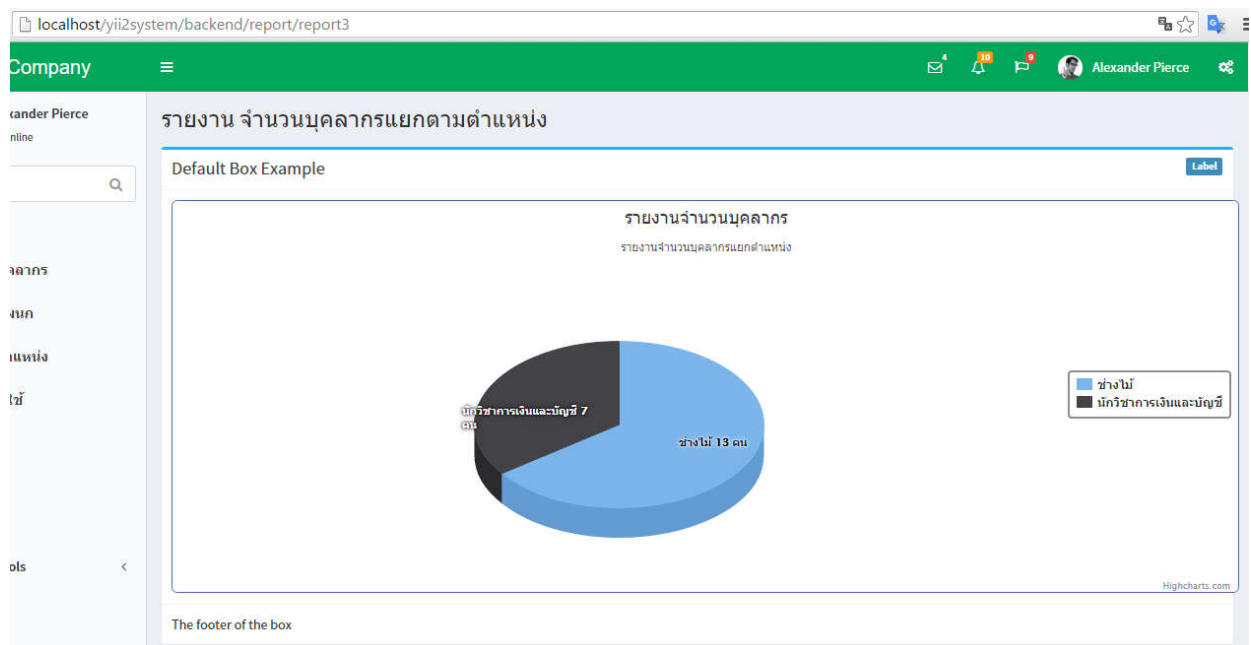
</div><!-- /.box -->

</div>

</div>

```

10.ทดสอบพิมพ์ Url เพื่อแสดงผล Pie Chart <http://localhost/yii2system/backend/report/report3>



อธิบายเพิ่มเติม เกี่ยวกับ Charts

หลักๆ แล้วโครงสร้างของ Charts ประกอบด้วย

**chart** เป็น top-level ของการตั้งค่า chart เช่น layout, event เป็นต้น

**series** คือ ชุดข้อมูล array มีทั้งแบบ single และ multiple

**xAxis/yAxis** คือ แกน x และ แกน y ของ charts มีคุณสมบัติที่สำคัญ เช่น labels, styles, backgrounds เป็นต้น

**title/subtitle** การกำหนดข้อความ title และ subtitle ให้กับ charts

**plotOptions** กำหนดคุณสมบัติของการ plot charts

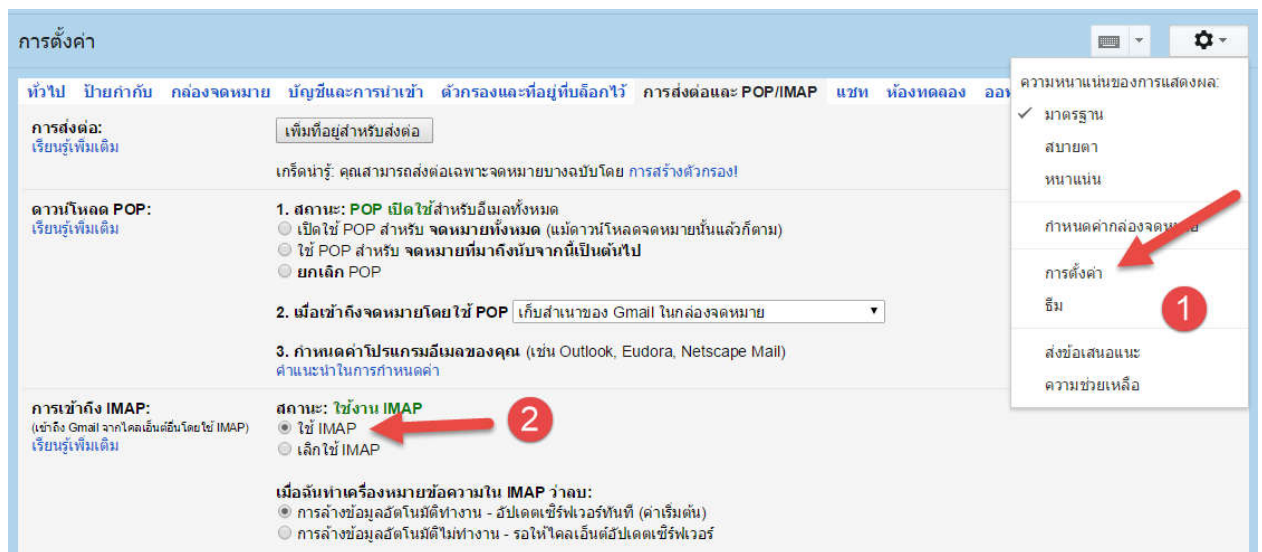
Note: ดูรายละเอียดทั้งหมดได้ที่ <https://github.com/miloschuman/yii2-highcharts>

และดูคู่มือการใช้งานแบบเต็มๆ ของ Highcharts ได้ที่ <http://www.highcharts.com/>

## บทที่ 4: โบนัสพิเศษ

### การตั้งค่าและทำงานกับอีเมล ด้วย SMTP

การส่งเมลแบบ SMTP นั้นในหนังสือเล่มนี้จะใช้ของ gmail เป็นหลัก หรือ เรียกอย่างหนึ่งว่า IMAP ซึ่งแนะนำให้ไปเปิดการตั้งค่าในส่วนนี้ก่อน



เมื่อเปิดเรียบร้อยแล้ว ต่อไปหากเราต้องการส่งเมลใน Yii 2 เราต้องไปตั้งค่า mailer component โดยให้เปิดไฟล์ common/config/main-local.php เพื่อตั้งค่าการส่ง ดังนี้

```
'mailer' => [  
    'class' => 'yii\swiftmailer\Mailer',  
    'viewPath' => '@common/mail',  
    // send all mails to a file by default. You have to set  
    // 'useFileTransport' to false and configure a transport  
    // for the mailer to send real emails.  
    'useFileTransport' => false,  
    'transport' => [  
        'class' => 'Swift_SmtpTransport',
```

```

'host' => 'smtp.gmail.com',
'username' => 'กรอกชื่ออีเมล gmail ของคุณเอง',
'password' => 'กรอกรหัสผ่าน gmail ของคุณเอง',
'port' => '587',
'encryption' => 'tls',
]
],

```

การตั้งค่าในส่วนของ username และ password ให้กำหนด username เป็นอีเมล และรหัสผ่าน ของแต่ละคนครับ

## ทดลองส่งเมล ด้วย SMTP

ถ้าเราสังเกตจะเห็นว่าในส่วนของ frontend นั้นจะมีหน้าฟอร์มสำหรับให้ผู้ใช้ส่งเมลติดต่อกับผู้ดูแลระบบอยู่ ถ้าล็อกอินอยู่ ลองล็อกเอาท์ แล้วเปิด Url ตามนี้ <http://localhost/yii2system/site/contact> จะเห็นได้ว่ามีฟอร์มติดต่อเราให้เรียบร้อย

ส่วน method สำหรับการส่งเมลนั้นจะอยู่ที่ไฟล์ frontend/models/ContactForm.php ลองเปิดแล้วเลื่อนลงมาล่างสุดครับจะเห็นว่ามี method contact() อยู่ให้แก้ไขได้ดั่งนี้

```

55 public function contact($email)
56 {
57     return Yii::$app->mailer->compose()
58         ->setTo($email)
59         ->setFrom([$this->email => $this->name])
60         ->setSubject($this->subject)
61         ->setReplyTo($this->email)
62         //->setTextBody($this->body)
63         ->setHtmlBody($this->body)
64         ->send();
65 }

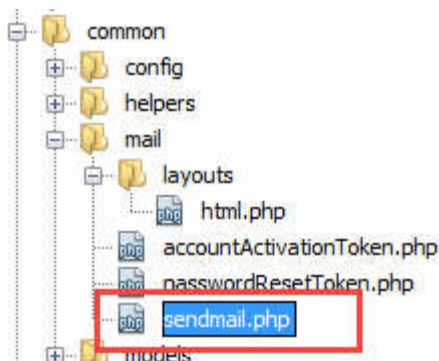
```

และถ้าหากตั้งค่าถูกต้อง เราสามารถทดสอบการส่งเมลได้จากตรงนี้ได้เลยครับ 😊

## การสร้าง email layout และ view สำหรับส่งเมล

ในบางครั้งเราอยากจัดรูปแบบอีเมลสวยๆ (ส่งแบบ HTML) หรืออยากจัดให้อยู่ในแบบที่เราต้องการ เราสามารถสร้าง layout และสร้าง view ได้ คล้ายกับ layout กับ view ในแบบปกตินั่นเอง แต่การสร้างเราต้องสร้างที่โฟลเดอร์ common/mail เท่านั้นครับ ขั้นตอน มีดังนี้

1. หากต้องการสร้างหรือแก้ไขไฟล์ layout ของเมล สามารถแก้ไขได้ที่ common/mail/layouts ลองเปิดดูที่ไฟล์ html.php ได้ครับ
2. ทดลองสร้างไฟล์ view ขึ้นมาใหม่ ในที่นี้ผมจะตั้งชื่อว่า sendmail.php โดยให้สร้างใน common/mail/



3. เปิดไฟล์ sendmail.php เขียนโค้ด ดังนี้

```
<h1>ข่าวประชาสัมพันธ์</h1>
```

```
<p>
```

```
เรียนคุณ <?= $name; ?> แจ้งเว็บไซต์ใหม่ เข้าได้ที่
```

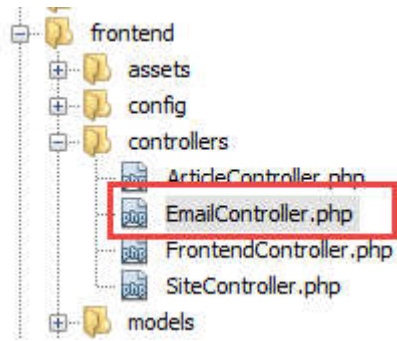
```
<a href="http://www.codingthailand.com">คลิกที่นี่</a>
```

```
<br>
```

```
ขอบคุณค่ะ
```

```
</p>
```

4. จากนั้นให้สร้างไฟล์ใหม่ชื่อว่า EmailController.php ที่โฟลเดอร์ frontend/controllers/



5. เปิดไฟล์ EmailController.php เขียนโค้ดดังนี้

```
<?php
```

```
namespace frontend\controllers;
```

```
use yii\web\Controller;
```

```
use Yii;
```

```
class EmailController extends Controller {
```

```
    public function actionIndex() {
```

```
        $this->sendMail('akenarin.k@ubu.ac.th', 'เอกนรินทร์ คำคุณ', 'ข่าวใหม่', 'data.xlsx');
```

```
        return $this->redirect(['site/index']);
```

```
    }
```

```
    public function sendMail($to, $nameTo, $subject, $filename) {
```

```
        $adminEmail = Yii::$app->params['adminEmail'];
```

```
        Yii::$app->mailer->compose('sendmail', ['name' => $nameTo]) //ส่งตัวแปร name ไปที่ view sendmail.php
```

```
        ->setTo($to)
```

```
        ->setFrom([$adminEmail => 'ข่าวสารจากเว็บ codingthailand'])
```

```
        ->setSubject($subject)
```

```
        ->setReplyTo($adminEmail)
```

```
        ->attach(Yii::getAlias('@webroot').'/uploads/'.$filename) //แนบไฟล์
```

```
        ->send();
```



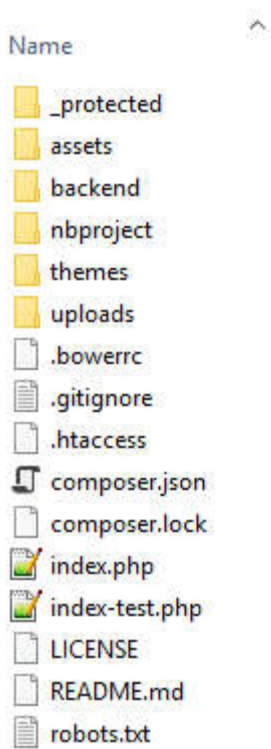
```
}  
  
}
```

6.ทดสอบส่งเมลโดยพิมพ์ที่ Url <http://localhost/yii2system/email/index/>

จากนั้นตรวจสอบที่เมลว่าได้รับหรือไม่ครับ

## แนะนำการอัปเดต Yii2 เพื่อใช้งานจริง

การอัปเดตขึ้น host ให้เราตรวจสอบก่อนว่า เวอร์ชัน PHP เป็น 5.4 ขึ้นไปหรือยัง จากนั้นเราก็ใช้โปรแกรม เช่น WinSCP เพื่ออัปเดตไฟล์ขึ้นไป โดยให้อัปเดตไฟล์ทั้งหมดนี้



หลังจากอัปเดตเรียบร้อยแล้ว ให้ chmod โฟลเดอร์ 2 โฟลเดอร์ได้แก่ **assets** และ **backend/assets** เป็น 777

อย่าลืม import ฐานข้อมูลด้วยนะครับ และกำหนด username, password ของฐานข้อมูลให้ถูกต้องด้วย

เรียบร้อยแล้วครับ สำหรับการเอาขึ้น Hosting

ในเวอร์ชัน 1 นี้ขอจบเนื้อหาเพียงเท่านี้ก่อน ยังมีเรื่องที่ยากเพิ่มเติมให้อีก ไว้จะแจ้งให้ทราบอีกครั้งหนึ่ง ขอให้สนุกกับการเรียนรู้ครับ

โค้ชเอก